

Robust Localization in Urban and Natural Environments

Abstract

This thesis proposes methods for global localization in urban and natural environments. The proposed approaches focus on segment-based localization systems for loop closures in the context of SLAM, alleviating the need for explicit fine-tuning or parameter fitting.

Initial analysis of current state-of-the-art systems are conducted and their performance in urban and natural environments is evaluated. We propose a hybrid feature descriptor method that does not rely only on the geometry of objects, but also considers the distribution of points within those objects. The segmentation differences encountered in natural environments are addressed by a novel key pose extraction technique.

Inspired by the ability of deep networks to generalize to multiple terrains, we propose a novel deep architecture capable of generalizing between natural and urban scenes. We design a state-of-the-art point cloud based network with efficiency in mind that is also deployable on computationally constrained platforms, such as drones, legged robots, or UGVs. The proposed objective function recognizes similar objects without any semantic information. We also extract a matchability score between segments that we use in a probabilistic geometric verification stage for coarse registration.

The analysis of the point-based architectures led to exploration of the bottlenecks in the computational complexity of these models. We propose a sub-sampling method allowing us to shrink an existing network architecture without sacrifice in performance. In addition, substantial analysis are presented on different training and test data variation for all the localization methods. The evaluation was performed on data from over 300 km of driving, that we manually pre-processed and made available to the community.

The final contribution is a method for learning keypoint regions that can be applied to any differentiable descriptor. The proposed framework is a substitute for the first step in segment-based localization systems.

This thesis also presents two applications of the above contributions. First, motivated by the expressive power of the geometric features, we propose a verification system that seeks to predict when localization failures occur in the environment. Specifically, we leverage some of the geometric features from our localization system to propose a classifier that predicts alignment risk based on overlap and degenerate geometric constraints. Second, we encapsulate the proposed loop closure method into a full SLAM system which is also deployed on the ANYmal quadrupedal robot. The proposed deep learned loop closure method shows improved scalability when compared to geometric-based loop closure systems.

Robust Localization in Urban and Natural Environments

D. Phil. Thesis

Oxford Robotics Institute
Department of Engineering Science
University of Oxford



Supervisor:
Doctor Maurice Fallon

Georgi Tinchev

Balliol College
Trinity 2020

Declaration

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. I declare that this thesis is entirely my own work, and except where otherwise stated, describes my own research.

Georgi Tinchev, Balliol College

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr. Maurice Fallon, for the constant nudge to keep me going and improving myself. Without his scrutiny none of the work would have been possible. I would also like to express my sincerest gratitude to the post-doctoral fellows I have been working with - i) Dr. Adrian Peñate-Sánchez for being the guy in my corner and fighting my battles constantly, ii) Dr. Milad Ramezani, Dr. Mathieu Geisert, and Dr. David Surovik for the unconditional support, fun lunches in Balliol, out-of-work activities, and online games during the quarantine. Other work colleagues with whom I have collaborated or without whom I would not have even written my first paper - Dr. Simona Nobili, Oliver Groth, Jonathan Attias, Tom Dobra, and everyone else at the Oxford Robotics Institute. I would also like to express my appreciation to the two interns that worked on this project and kept improving my knowledge - Egor Iuganov and Jakub Perlin. Also, I would like to thank Dr. Thomas Drugman for giving me a chance and constantly going beyond his responsibilities. Also Dr. Marius Cotescu and Dr. Yunlong Jiao for navigating me through the dark paths of speech synthesis and machine learning.

Secondly, I would like to thank my parents for the continuous support they have provided throughout these years of uncertainty, stress, and depression. Mainly to my grandparents, Tinko and Bogdanka, whom I both lost during the span of this venture. Without them this would not have been possible. Thank you for giving me a hand when even my parents were resisting. Thank you for the help in times of need and pushing me to greater limits. Thank you for constantly helping me fight my battles and making this reality. I owe you everything.

Finally, I would like to thank my girlfriend, Natalia, who supported me in the final months of my DPhil and was keeping up with me in the first place. Also, I would like to thank the Oxford University Volleyball club for boosting my competitive side and the thousands unforgettable moments. I would like to thank my friends - Nikola Petkov and Konstantin Gizdov for showing me the light in the darkest hours of this journey. I would also like to express my gratitude to Neil Stirton who jump started my career path and stirred me in the right direction from the very beginning. Also, thanks to all the proofreaders and to you for reading this.

Contents

1	Introduction	1
1.1	Objective	1
1.2	Thesis outline	4
1.2.1	Localization systems in challenging environments	5
1.2.2	Efficient deep learning for place recognition	6
1.2.3	Generalization and large scale evaluation	7
1.2.4	Descriptor-agnostic keypoint detection	9
1.2.5	Prediction of alignment risk failure	10
1.2.6	Real-time application for loop closures	11
1.3	Contributions	11
1.4	Relevant publications	14
2	Background	16
2.1	LIDAR Odometry	17
2.1.1	Iterative Closest Point	17
2.1.2	Alternative LIDAR Odometry methods	19
2.1.3	Predicting registration failure	20
2.2	Localization	21
2.2.1	Localization in Robotics	22
2.2.2	LIDAR localization in Robotics	23
2.2.3	Segment-based LIDAR Localization in Robotics	24
2.2.4	Segment-based localization problem statement	26
2.3	Feature learning from 3D data	28
2.3.1	Voxel-based approaches	29
2.3.2	Point-based approaches	30
2.3.3	Other 3D learning approaches	32
2.4	Keypoint extraction	33
2.4.1	Extracting keypoints in images	33
2.4.2	Extracting keypoints from 3D data	35
2.5	Summary	38
3	Seeing the Wood for the Trees: Reliable Localization in Urban and Natural Environments	40

4	Learning to See the Wood for the Trees: Deep Laser Localization in Urban and Natural Environments on a CPU	51
5	Real-time LIDAR localization in natural and urban environments	62
6	SKD: Keypoint Detection for Point Clouds using Saliency Estimation	93
7	Predicting Alignment Risk to Prevent Localization Failure	104
8	Online LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure	115
9	Future work & Conclusions	125
	9.1 Conclusion	129
	References	130

List of Abbreviations

BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DoF	Degrees of Freedom
EKF	Extended Kalman Filter
FAST	Features from Accelerated Segment Test
FMCW	Frequency-Modulated Continuous Wave
GMM	Gaussian Mixture Model
GPU	Graphical Processing Unit
ICP	Iterative Closest Point
ISS	Intrinsic Shape Signatures
IMU	Inertial Measurement Unit
IEEE	Institute of Electrical and Electronics Engineers
KNN	K Nearest Neighbor
LIDAR	Light Detection and Ranging
NARF	Normal Aligned Radial Feature
ORB	Oriented FAST and rotated BRIEF
PCA	Principal Component Analysis
PROSAC	Progressive Sample Consensus
RANSAC	Random Sample Consensus
RSJ	Robotics Society of Japan
SLAM	Simultaneous Localization and Mapping
SIFT	Scale Invariant Feature Transform
SURF	Sped Up Robust Features
UGV	Unmanned Ground Vehicle
VO	Visual Odometry

1

Introduction

1.1 Objective

Understanding where a robot is and what surrounds it are fundamental capabilities necessary to achieve autonomy. Formally known as Simultaneous Localization and Mapping some state-of-the-art approaches include metric methods, such as the works of Davison et al. [2007], Newcombe et al. [2011], and Mur-Artal et al. [2015], topological-based such as the methods proposed by Ulrich and Nourbakhsh [2000], Cummins and Newman [2008], and Milford and Wyeth [2012], and semantic-based methods from Castle et al. [2007], Civera et al. [2011], and Cadena et al. [2015]. These approaches have achieved tremendous results over the past 20 years. The work proposed in this thesis aims to complement such approaches by evaluating the localization algorithms in challenging environments, such as forests, parkland scenery, and industrial sites.

There are two predominant approaches when building a localization system - direct and feature-based methods (fig. 1.1). Direct methods such as [Newcombe et al. 2011; Pizzoli et al. 2014] operate on raw pixels and work remarkably well in texture-less scenarios. Feature-based methods describe the surroundings in a compact representation that captures different qualities of a region - color, intensity, texture, contrast, and distribution of points among others. Well known

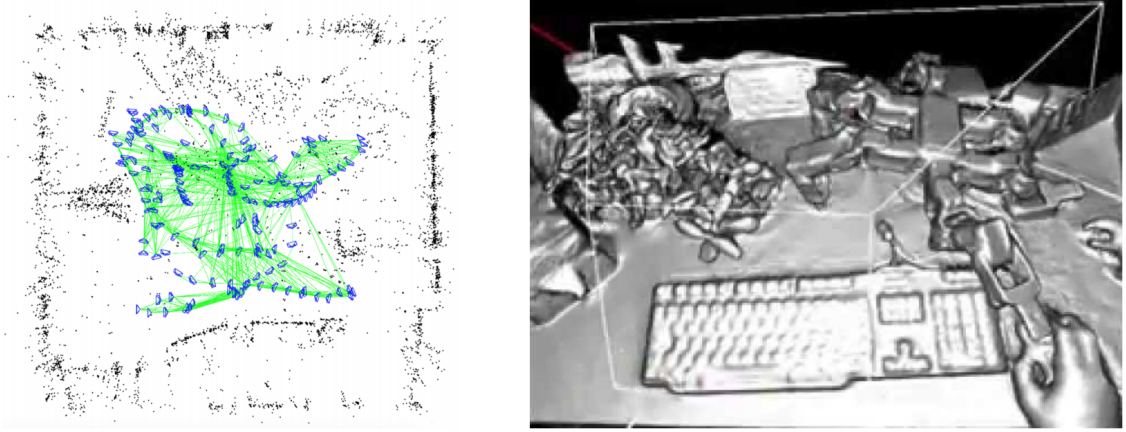


Figure 1.1: Example SLAM systems. Left: feature map of a room produced by ORB-SLAM [Mur-Artal et al. 2015]. Right: dense map of a desktop from DTAM [Newcombe et al. 2011]. Image adopted from [Cadena et al. 2016].

examples of hand-engineered features in the image domain include SIFT [Lowe 2004], SURF [Bay et al. 2006], ORB [Rublee et al. 2011], BRISK [Leutenegger et al. 2011], and BRIEF [Calonder et al. 2010]. These advanced the state-of-the-art in feature-based methods tremendously over the past decade. A key objective of this thesis is to evaluate the generalization ability of features extracted from 3D data to multiple environments.

Modern advances of deep learning achieved exceptional visual recognition results. The works of Sharif Razavian et al. [2014], Oquab et al. [2014], and Arandjelovic et al. [2016] are examples of solid fundamentals in the field. In the image domain, these learning techniques surpass the descriptive power of engineered features [Simo-Serra et al. 2015]. Recently, deep learning was adopted to efficiently support 3D data as input [Qi et al. 2017a; Angelina Uy and Hee Lee 2018; Wang et al. 2019]. A common problem with deep learning is the need of a GPU during operation, which is not always possible on different types of robots such as drones, quadrupeds, or Unmanned Ground Vehicles (UGVs). This thesis aims to design an efficient solution suitable for deployment on robots with limited computational power.

Feature-based methods for localization can be split in two parts - detecting the features and recognizing them in different observations. The former is also

known as *keypoint extraction*, while the latter, which describes features with a fixed length *descriptor* vector, is known as *feature description*. There has been a lot of research in recent years on both topics, however, fundamental knowledge about what defines an area of interest is still an open research question. One of the objectives of this thesis is to seek understanding of what defines a specific location as salient for a given descriptor.

Robustness of localization systems is not a well studied topic. The absence of geometric features can cause SLAM systems to drift and produce incorrect localizations. One such example is when navigating long corridors of a building. In this scenario the lack of structure in one of the dimensions makes it hard for localization systems to function. This work analyzes features that are not only suitable for localization but can also aid other systems. A secondary objective of this thesis is to provide an additional application domain for geometric features, such as the one capable of preventing a localization failure before it happens.

Nowadays, deep learning methods often constrain themselves on evaluation with standard metrics and datasets [Dosovitskiy et al. 2015; Eigen et al. 2014; Žbontar and LeCun 2016]. However, it is important to also tackle challenges presented by integrating these types of systems on an actual robot and deploying them in realistic scenarios. Examples of these challenges include designing the code structure into easily extendable individual self-sufficient modules, interaction between these modules, deployment of proof-of-concept learning models to real-time operation. The final aim of this thesis is to create a localization system that can be deployed as part of a full SLAM module running on a robot in real time.

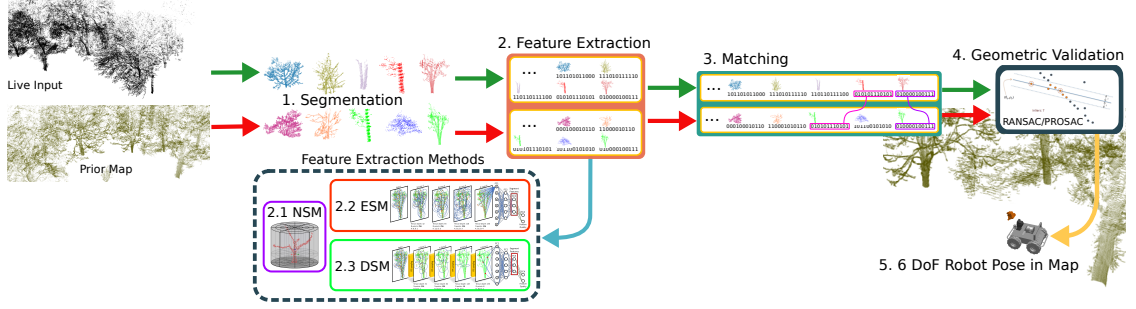


Figure 1.2: Global localization pipeline. This thesis focuses on improving individual modules of this pipeline.

1.2 Thesis outline

A brief overview of the thesis content is presented in this section. Figure 1.2 introduces individual modules of the global localization approach that this thesis improves upon.

We start by overviewing existing **localization systems on both urban and natural environments**. Driven by our findings, we engineered features suitable for different natural environments such as parkland, forested, and urban scenes (module 2.1 in fig. 1.2). The following sections shift to exploring a novel **efficient deep learning architecture for place recognition** capable of obtaining learned features during inference time without the need of a GPU (module 2.2 in fig. 1.2). Motivated by our findings, the ensuing section dives deeper into **generalization and large scale evaluation** of the proposed model. Our analysis indicates that we can further reduce the computational load of the algorithm by sub-sampling the number of points in the inner layers of the architecture (module 2.3 in fig. 1.2). We also examine a variety of different LIDAR sensors and evaluate the generalization ability of our methods with different training/testing data combinations. Driven by these results, the next section proposes an alternative to the segmentation module in segment-based localization approaches – a deep learning method for **descriptor-agnostic keypoint detection** (module 1 in fig. 1.2)). The proposed approach learns to predict places of interest which can be derived from any differentiable descriptor. By directly optimizing for matchability and repeatability of keypoints, the proposed approach outperforms existing state-of-the-art

methods. Motivated by the feature design in our previous contributions, the following section proposes to use geometric features, similar to the ones used in natural environments, to **predict alignment risk failure** (module 4 in fig. 1.2). The method boosts the performance of a classifier capable of predicting when a localization failure may occur. The final contribution proposes a **real-time application for loop closures** by deploying the localization algorithm as part of a real-time SLAM system on a quadruped robot traversing an industrial site (modules 1-5 in fig. 1.2).

1.2.1 Localization systems in challenging environments

The main goal of this thesis is to produce a reliable system for both urban and natural environments. While the goal seems well motivated, it is important to first analyze the capabilities of existing state-of-the-art approaches and their generalization ability to more challenging, natural datasets. This provides valuable insights into specific localization modules that need additional research.

In chapter 3 we start by evaluating the currently existing state-of-the-art approaches on three datasets - urban, parkland, and forested scenery. Previous approaches, suitable for operation in urban environments, either rely on matching local features in the environment [Rusu et al. 2009; Steder et al. 2010b], describing the entire environment with one descriptor vector [Magnusson et al. 2009; Granström et al. 2011; Röhling et al. 2015], or a combination of both [Douillard et al. 2012; Nieto et al. 2006]. Local feature-based approaches often lack descriptive power, while global ones struggle with collision variation and invariance to different conditions. Therefore, we opt for a combination of both - segment the environment in a collection of objects and describe each object individually, similar to the work of Dubé et al. [2017].

We observed that it is challenging to determine a representative point for each object in natural environments, due to interleaved trees or even just windy conditions. Therefore, we propose a strategy to mitigate this by orienting each segmented object to its principal direction. This allows us to extract a consistent

keypoint which improves the description performance. We revisited geometry-based and shape-based features and found that there is room for improvement for both urban and natural environments. Thus, we proposed a hybrid feature descriptor that is not based only on the geometry of the segment, but also on the distribution of points with respect to the keypoint. The key insight of this contribution is that natural data are much more challenging than urban data. It is also unclear whether we can create a universal feature descriptor that can achieve similar localization performance in both environments.

1.2.2 Efficient deep learning for place recognition

Motivated by the results in chapter 3 we focus on designing features that will work in both forested and urban environments. Inspired by the recent advances of deep learning in image object classification [Krizhevsky et al. 2012; Simonyan and Zisserman 2014; He et al. 2016], we evaluated approaches suitable for 3D data [Qi et al. 2017a; Qi et al. 2017b; Li et al. 2018]. Our contributions shows that per-point Multilayer Perceptron-based (MLP) models not only push the boundaries of object recognition in these challenging environments, but also do so without the need of a GPU during operation.

Due to the nature of our system, the segmentation stage does not guarantee semantically meaningful information. This is desired, as in a forest there would be at most a few special semantic classes. Instead, it provides individual objects that can be matched between observations. In order to train a deep learning system on a non-standard dataset, one needs to label individual examples. This is especially challenging for natural data, due to the similarity of objects in these environments. Therefore, we leveraged the motion of the vehicle to automatically segment and label objects in consecutive LIDAR scans in order to train our system.

As a result of annotating similar and dissimilar objects in the environment, chapter 4 proposes a novel architecture that learns a feature descriptor based on an

objective function comprised of two terms - triplet and pairwise losses. The former is responsible for differentiating between dissimilar objects, while the latter clusters together similar objects in feature space. This allows us to learn a meaningful descriptor without the need for semantic classes, nor per-point annotations.

A desirable outcome was to train a deep learning system, capable of operating without the need of a GPU. Therefore, our network architecture was designed with only a few layers in mind. Due to the vanishing gradient problem [Wang et al. 2014] and the inability to use residual connections in shallow networks [He et al. 2016], it is not trivial to train such small models with the above objective function.

Global localization approaches typically compel themselves to apply RANSAC alignment once a set of correspondences is available [Elbaz et al. 2017; Rusu and Cousins 2011]. In chapter 4 we go beyond simple geometric constraints and leverage a probabilistic method for registration based on PROSAC. Therefore, we propose a classifier branch to the network that predicts the quality of two descriptors. The network branch is responsible for producing a probability of how likely the two descriptors are a match. The probability is directly used as a filter to remove incorrect segment correspondences during the PROSAC registration stage.

An important empirical observation is that features trained with the model perform much better in these challenging natural conditions. Furthermore, per-point MLP-based models with shallow architecture design allow the use of a CPU during inference, which is rare in deep learning methods.

1.2.3 Generalization and large scale evaluation

Deep learning methods often do ablation studies for specific method to explore which part of the model contributes the most to the end result [Girshick et al. 2014; Uijlings et al. 2013]. Driven by the architectural choice of chapter 4, we decided to perform ablation studies in order to determine the computational

bottleneck of the architecture. Our analysis indicate that the slowest part is the number of points processed in each layer of the model. Therefore, in chapter 5 we propose a method that gradually reduces the number of points in each subsequent layer via subsampling. We propose a strategy to extract points further away from each other, in order to keep the feature space as diverse as possible for the first few layers of the architecture.

Chapter 5 follows the main theme of this thesis — providing an algorithm for robust localization in both urban and natural environments. Given the methods in chapters 3 to 5, it is not straightforward to derive whether a model needs to be retrained when used in a different environment. Therefore, we analyze the generalization abilities of all the aforementioned methods with different permutation of train and test data between forested and urban scenarios. Furthermore, we also expose a common issue associated with rotation invariance when processing 3D point cloud data. We propose a simple alternative that does not embed any explicit rotation invariance within the network, but rather operates at the data preprocessing stage. This is important since we do not want to compromise performance, while also being robust to viewpoint and training data variation.

Deep learning often relies on large scale datasets to enable autonomous driving research progress. Renowned datasets such as the KITTI [Geiger et al. 2013] or the Oxford RobotCar [Maddern et al. 2017] currently provide GPS and GPS-INS ground truth data. While this is useful to do global point cloud scene matching, it is insufficient for precise per-point association. Since supervised deep learning often relies on accurate annotation of large scale datasets, we post-processed 30 sequences of the Oxford RobotCar dataset to create one of the largest labeled LIDAR dataset. Two example scenes of the proposed dataset are depicted in fig. 1.3.

A key observation in chapter 5 is that sampling in the inner layers of per-point MLP-based models reduces dramatically the computational load of the

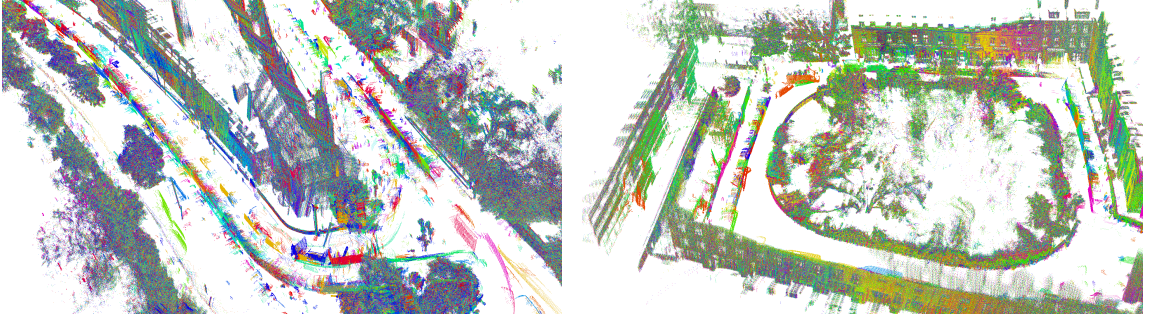


Figure 1.3: Examples of the Aligned RobotCar LIDAR dataset. Post-processed 30 sequences of the RobotCar dataset [Maddern et al. 2017]. Left: Parks Road, Oxford. Right: Wellington Park, Oxford.

system, which results in quicker inference speeds and lower memory consumption at negligible performance cost. In addition, we were pleased to see that the proposed models generalize incredibly well between the environments with at most 1-2% drop in performance. This suggests that a model trained on a standardized dataset can be directly deployed in natural environments.

1.2.4 Descriptor-agnostic keypoint detection

The work presented in chapter 6 follows the main goal of this thesis by focusing on the first part of the localization pipeline — finding repeatable places of interest in the environment. This is essential for feature-based localization methods, as it is not trivial how to detect descriptive and informative landmarks.

Recent research has started exploring the possibility to learn keypoints directly from data. Usually the keypoints are learned simultaneously with the descriptor vector [Yew and Lee 2018; Li and Lee 2019]. However, this often leads to sub-optimal results due to the pre-training the descriptor first in order to obtain initial keypoints. In chapter 6, we extract the saliency information of a descriptor in order to detect keypoint candidates. The method then refines the initial set of candidates based on a combination of geometry, context information, and saliency score to produce a plausible set of keypoints.

The key outcome of chapter 6 shows that keypoint generation and feature descriptor illustrate a chicken and egg problem. One needs the keypoints to

produce a feature vector, however, using the feature descriptor to generate keypoints results in higher repeatability between individual observations and higher probability to be matched by the descriptor. An additional benefit of the proposed approach is its applicability to any differentiable descriptor.

1.2.5 Prediction of alignment risk failure

The main goal of this project supports the overall objective of the thesis - producing a reliable localization system. To this end, we explored an extension to a typical localization system designed to minimize the risk of an incorrect registration between consecutive scene observations. Such a system ensures reliable operation with low failure rate for long periods of time and adapts to different challenging environments. Our contributions show that geometric features are an integral part of predicting when a localization will fail.

A prominent example of localization failure scenario when traversing indoor environments is to navigate a path between two rooms. In this case two consecutive 3D LIDAR observations of the two rooms will have very little in common to determine the motion of the robot. Similarly, when trying to localize within a long corridor without any distinctive features, the registration will likely fail. Motivated by this, chapter 7 proposes a system classifying when an alignment will likely fail between two scans of the environment. It is based on two features that describe the overlap between two observations and the general geometric properties of the environment. The approach was evaluated in a combination of indoor and outdoor scenarios as well as a small simulated dataset.

It is interesting to observe that the distribution of points in the environment can be expressed as a single feature which, in combination with the percent of overlap between consecutive clouds, can reliably detect when an alignment will fail. The main observation is that eigenvalue-based features, similar to the ones used for localization, can detect the degree to which alignment is affected by geometric instabilities.

1.2.6 Real-time application for loop closures

Regularly, academic research in computer vision and deep learning is terminated at the level of proof of concept, where algorithms are designed and evaluated against common datasets and benchmarks. Often times these algorithms do not have any immediate practical application. It is challenging to transfer models, built as a proof of concept, to a production-ready system. Following the main topic of this thesis, we surpass this notion and implemented our algorithm as a loop closure detector in a factor-graph based 3D LIDAR SLAM system.

Standard SLAM systems, such as [Engel et al. 2014; Campos et al. 2020] are usually comprised of two components — odometry estimation and loop closure detection. In chapter 8 the proposed SLAM system combines the work of [Nobili et al. 2017] with the work presented in chapter 4. To this end, we leveraged a modular approach to designing the system in which we can substitute the model’s weights effortlessly while keeping the input and the output untouched. The resulting system also leverages the proposed verification method from chapter 7. We analyze the computational complexity of the verification method and propose an improvement that enables real time operation.

One of the key contributions of chapter 8 is the evaluation of the system in an industrial site on a quadruped robot. This not only shows the generalization ability of the proposed deep learning model to unseen data, but also shows variation to different LIDAR sensor and mode of operation. The empirical findings indicate that the proposed learning method for loop closure detection scales better with large map sizes when compared to geometric-based loop closures.

1.3 Contributions

This section summarizes the contributions presented in this thesis.

Localization in natural environments Chapter 3 presents an approach dubbed *NSM* suitable for localization in both urban and natural environments. It

proposes a novel keypose extraction method more suited to natural environments. It uses a hybrid feature description method that improves on the state-of-the-art. The algorithm was evaluated on urban, foliage-heavy, and parkland environments in terms of descriptor quality as well as overall registration accuracy.

Computationally efficient localization Chapter 4 presents an extension of *NSM*, named *ESM* that learns a feature descriptor from data. The proposed architecture is designed to be efficient and deployed on robots without the need of a GPU. The method was trained with a combination of triplet and pairwise losses between anchor, positive, and negative segments without any semantic information. This improves the state-of-the-art for feature learning on both urban and natural environments.

Probabilistic Geometric Verification Chapter 4 also presents a probabilistic method for estimating the alignment between a current scene and a previous map. The method is a direct substitute for the last part of the pipeline where a list of correspondences and a probability of how likely two segments are a match are available. The probability is directly extracted from the descriptor network and shows superior performance to prior work.

Robustness and efficiency in LIDAR localization In chapter 5 we propose a novel strategy for reducing the computational complexity of per-point MLP-based neural networks, by gradually down-sampling the number of points in the inner layers of the model. The new architecture, named *DSM*, improves upon the state-of-the-art by having 3x quicker inference speed and consumes 70% less memory. We also evaluate the generalization abilities of our methods — *NSM*, *ESM*, and *DSM* — across multiple urban and natural scenes, as well as LIDAR sensor variations.

Large annotated LIDAR dataset We propose an annotated LIDAR dataset for accurate per-point association, which we provide for the general public. It consists of association between 30 sequences of the original Oxford

RobotCar dataset [Maddern et al. 2017]. We leverage this dataset in our analysis of the generalization abilities of each of our algorithms across multiple urban and natural scenes.

Learning keypoints instead of segments Chapter 6 presents an approach to learn reliable keypoints based on the saliency information of a differentiable descriptor w.r.t. the input. The algorithm is descriptor agnostic by design. The method was evaluated on standardized datasets and outperforms the state-of-the-art on matchability and repeatability metrics for keypoint generation. Interestingly, due to the consistent keypoint generation, the time taken for RANSAC to find an accurate transformation is reduced by a factor of two. This work can be used as a substitute of the segment generation module in a segmentation-based localization system, where a neighborhood of points close to the chosen keypoint can be selected for description.

Usefulness of geometric features We leverage the geometric features, similar to the ones presented in chapter 3, to predict when a registration between two observations will likely fail. The approach in chapter 7 uses the proposed eigenvalue-based feature as part of a classifier to estimate the risk if incorrect transformation when registering point clouds with the Iterative Closest Point algorithm.

Online SLAM system An important contribution was to develop the application of *ESM* on GPU-constrained robots in order to show real time performance. In chapter 8 we integrate *ESM* on the ANYMAL quadruped robot and demonstrate the performance of the learned loop closure algorithm in a real-time application. The proposed method scales better than metric loop closures with larger map sizes. We also extend the work presented in chapter 7 to achieve real time performance.

1.4 Relevant publications

Peer-reviewed publications that are relevant to the content of this thesis are listed below. Some publications have been submitted and are currently in review. The references to the corresponding thesis chapters are included as well.

Chapter 3

Tinchev, G. Nobili, S., Fallon, M.:

Seeing the Wood for the Trees: Reliable Localization in Urban and Natural Environments,
IEEE/RSJ International Conference on Intelligent Robots and Systems, 2018

Chapter 4

Tinchev, G., Penate-Sanchez, A., Fallon, M.:

Learning to See the Wood for the Trees: Deep Laser Localization in Urban and Natural Environments on a CPU,

IEEE Robotics and Automation Letters, 2019, Q1 JCR + IEEE International Conference on Robotics and Automation, 2019

Chapter 5

Tinchev, G. Penate-Sanchez, A., Fallon, M.:

Real-time LIDAR localization in natural and urban environments,

Submitted to Science Robotics 2020 Special issue: Deep Learning & Robotics,
Q1 JCR

Chapter 6

Tinchev, G. Penate-Sanchez, A., Fallon, M.:

SKD: Keypoint Detection for Point Clouds using Saliency Estimation,

Submitted to IEEE Robotics and Automation Letters, 2021, Q1 JCR + IEEE International Conference on Robotics and Automation, 2021

The following publications were done in collaboration with other students and post doctoral researchers. The first was published due to important contributions from chapter 3, while the second presents the applicability of the proposed method.

Chapter 7

Nobili, S., Tinchev, G., Fallon, M.:

Predicting Alignment Risk to Prevent Localization Failure,

IEEE International Conference on Robotics and Automation, 2018

Chapter 8

Ramezani, M., Tinchev, G., Iuganov, E., Fallon, M.:

Online LIDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure,

IEEE International Conference on Robotics and Automation, 2020

2

Background

The contents of chapters 3 to 8 describe our contributions and cover a variety of areas in robotics and computer vision. Here we will outline the relevant literature with respect to each individual chapter.

The main focus of this thesis is on global localization in challenging environments. We begin by outlining the basic concepts related to localization - odometry, loop closures, and SLAM. The first section is focused on LIDAR **odometry** and state-of-the-art methods for registration and predicting localization failure. The following sections dive deeper into the history of **localization** in different domains such as images, radar, sonar, and LIDAR. Specifically, this section provides the problem definition of segment-based localization system. The ensuing section focuses on **deep learning methods capable of handling 3D data** since we leverage them in our contributions. It revisits the recent history of methods that learn from point cloud data converted to voxels, raw point clouds, and other representations. The final section of this chapter reviews engineered and learned **keypoint extraction** methodologies in both images and point clouds.

In order to acquire the position of a robot in the environment, the robot needs to estimate its own motion, as well as to keep an internal representation of how the *world* looks like. In this thesis, we refer to *odometry* as the estimated relative motion of a frame using the current sensor measurements with respect to a fixed

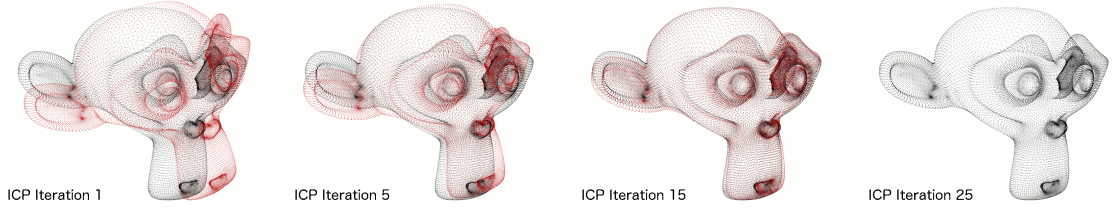


Figure 2.1: ICP Registration example of a monkey point cloud. The *reading* red point cloud aligned to the *reference* black point cloud at different iterations of ICP. Image adopted from [Rusu and Cousins 2011].

frame. In this context, *localization* is used to solve the problem of identifying where a sensor is located relative to a previously acquired map. Furthermore, *mapping* refers to the problem of combining multiple measurements into a single internal belief system of how the environment looks like. When localization is carried out concurrently with mapping without prior information it is often treated as a research field of its own - Simultaneous Localization and Mapping (SLAM) as reviewed by Cadena et al. [2016].

2.1 LIDAR Odometry

Odometry estimates the change in position over time of a sensor - typically a camera, radar, or a light detection and ranging (LIDAR). It uses the data from the sensor to estimate the motion. The following section follows the main topic of the thesis where we review methods that estimate LIDAR odometry in consecutive point cloud scans. We start by presenting the Iterative Closest Point algorithm and continue by describing state-of-the-art approaches for LIDAR odometry. In addition, chapters 7 and 8 present approaches that predict when two consecutive observations are unlikely to register. Therefore, the ensuing section provides an overview of different strategies to predict alignment failure.

2.1.1 Iterative Closest Point

The Iterative Closest Point (ICP) is a popular approach for 3D dense cloud registrations [Arun et al. 1987; Besl and McKay 1992]. Depicted on fig. 2.1, the

method aligns a *reading* cloud to a *reference* point cloud [Pomerleau et al. 2013]. It assumes an initial estimate for the pose of the *reading* point cloud. The algorithm can be split in four stages: i) pre-filtering, where ICP discards redundant points, extracts local features and computes descriptors (most commonly normals); ii) data association, where pairs of points from the two clouds are matched based on Euclidean distance; iii) outlier rejection, which removes points without correspondences; and iv) error minimization, where a distance metric is minimized to bring the point clouds into alignment after a number of iterations. There are two notable strategies for minimizing the error: point-to-point [Besl and McKay 1992] and point-to-plane [Chen and Medioni 1992]. The work of Nüchter et al. [2007] is one of the pioneering works that took ICP scan matching to a full 6D SLAM method by extrapolating the odometry, providing initial pose estimate from an octree, loop closing, and cached K-D search.

Biber and Straßer [2003] introduced an alternative to the classic ICP approach, named the Normal Distribution Transform (NDT). It defines normal distributions on a grid that are later matched to any smooth cell between the *reading* and the *reference* clouds. Thereafter, standard optimization algorithms are used to estimate the alignment. In comparison to ICP, NDT provides a larger basin for convergence, but is less predictable [Magnusson et al. 2007].

Segal et al. [2009] combined the point-to-point and point-to-plane metrics into a single approach named Generalized ICP (GICP). The authors introduced a novel plane-to-plane strategy between the *reference* and *reading* point clouds. The error minimization was defined in a probabilistic way w.r.t. a Gaussian distribution, centered at the point of perfect correspondence.

Serafin and Grisetti [2015] combined the feature representation of the points with the 3D-NDT approach [Magnusson et al. 2007]. The algorithm, named Normal-based ICP, sought to match normal features extracted from a Gaussian cell in the *reference* cloud to any cell in the *reading* cloud. The error minimization function optimizes the distance between point pairs and their normals.

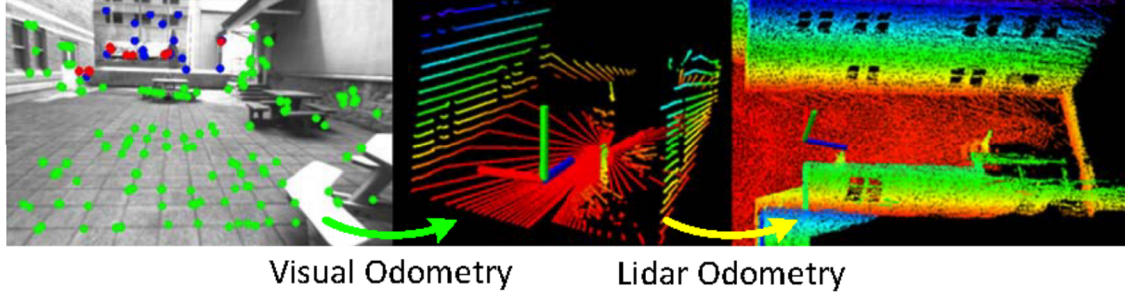


Figure 2.2: Example of motion estimate using visual (left) and LIDAR (right) odometry. Image adopted from [Zhang and Singh 2015].

Nobili et al. [2017] presented an ICP approach which analyzed the overlap between two clouds, named Autotuned ICP (AICP). AICP automatically adjusted the outlier filtering based on the overlap between the *reading* and the *reference* point clouds. The approach was evaluated on humanoid robots in a laboratory environment and datasets from the DARPA Robotics Challenge Finals.

2.1.2 Alternative LIDAR Odometry methods

Next, we focus on alternatives to ICP that also estimate the egomotion given raw LIDAR sensor measurements.

Zhang and Singh [2014] introduced LOAM, which is the most popular LIDAR odometry algorithm developed to date. The algorithm reliably tracked edges and corner features in consecutive LIDAR scans. It computed point-to-line and point-to-plane distances between corresponding edge and planar points respectively. Afterwards, they leveraged the Levenberg-Marquardt [Hartley and Zisserman 2003] method to solve the LIDAR motion. Its follow-up work, VLOAM (fig. 2.2), combined visual information and achieved state-of-the-art results on the KITTI dataset [Zhang and Singh 2015; Geiger et al. 2013]. Their loosely coupled system assumed zero velocity when using an inertial measurement unit (IMU) measurements and used them as a prior to the LIDAR system.

Ye et al. [2019] improved LOAM by introducing a tightly coupled method that *pre-integrated* the raw IMU measurements with the LIDAR scans directly. Their method, named LIOM, worked better in degraded scenarios and under

rapid motion. Since the authors proposed to use all the measurements to perform a joint optimization, LIOM failed to achieve real-time operation.

Most recently, Shan et al. [2020] improved on LOAM and LIOM by implementing a tightly coupled approach with a factor-graph representation. Their approach, named LIO-LOAM, leverages a local scan-matching algorithm that matched features of the most recent scan to a selected subset of keyframes. This allowed them to run in real time due to the sparse factor graph representation of only a few keyframes.

2.1.3 Predicting registration failure

In the following paragraphs, we introduce different approaches to predict whether an alignment between the *reading* and the *reference* point clouds will fail. Part of our contribution, presented in chapters 7 and 8, relates to registration verification. It is important to note that registration is not always possible between a pair of observations, and in that scenario a system should be robust enough to detect it.

Robustness in perception problems is not a well studied topic. In the context of SLAM it is commonly handled via a heuristic. For instance, Hu et al. [2012] uses two algorithms to estimate the motion and switches between them when one of them fails. In terms of optimization, the Lavenberg-Marquart method is a well-known alternative to the Gauss-Newton method that is more robust in terms of finding global optimum [Hartley and Zisserman 2003; Nocedal and Wright 2006]. Next, three parameters for predicting ill-constrained solutions of state estimation systems are discussed.

Zhang et al. [2016] introduced a novel parameter, named *degeneracy*, which estimated the degree to which an optimization-based state estimation system would fail in certain directions of the state space. It analyzed the geometric structure of the problem constraints in order to mitigate problems in the optimization. It then separated the state space, and only solved the problem in well-defined

directions, while using a best-guess estimate in the degenerate directions. Degeneracy was estimated after the point-wise data association step of ICP, and was therefore highly sensitive to initialization.

Given a linearized system $\operatorname{argmin}_x \|\mathbf{A}\mathbf{x} - \mathbf{b}\|$, we can compute the covariance of the linear solution as $\Sigma = \sigma^2(\mathbf{A}^T \mathbf{A})^{-1}$, where the variance computed from the residuals is defined as $\sigma^2 = \|\mathbf{A}\mathbf{x}_0 - \mathbf{b}\|^2 / (n - m)$. Here, n and m correspond to the number of constraints and the dimension of the linear problem. The *Inverse Maximum Covariance Eigenvalue* (IMCE) is defined as the inverse of the maximum eigenvalue of Σ [Trefethen and Bau III 1997].

The *Inverse Conditional Number* (ICN) of a linear system is another way to determine whether a linear system is ill-conditioned. The conditional number [Cheney and Kincaid 2012] was computed by the ratio of the minimum and maximum eigenvalues of $\mathbf{A}^T \mathbf{A}$. A large number indicated an ill-conditioned system. Similarly to degeneracy, ICN was also sensitive to initialization.

2.2 Localization

The main theme of the thesis is focused on localization. In this context, the main contributions of chapters 3 to 5 and 8 are related to global localization. In the following section we detail the background for general localization approaches, LIDAR localization, and segment-based LIDAR localization.

Localization is the problem of determining the position of a current observation with respect to a prior map. Formally, localization is defined as estimating the relative transformation T of a scene S_t at time t , to the prior map M defined as ${}^M T_t^S$. When doing LIDAR localization, S_t can be comprised of a single 3D scan of the environment, or a collection of vertical 2D or 3D scans between times $t - k$ and t . In this context, the term *loop closure* refers to recognizing a previously visited location. Typically, loop closure algorithms compute a measure of similarity between different observations and update the beliefs of a common system. The key problem of loop closure detection is often treated as being equivalent to global localization as evidenced by Granström et al. [2011].

2.2.1 Localization in Robotics

Image localization is the most widely explored approach to this problem, as evidenced by Furgale and Barfoot [2010], Churchill and Newman [2013], and Agarwal et al. [2011] and is also the approach with the most discriminative ability. Image-based localization can work with low cost cameras — even in the absence of physical structure. In comparison to RADAR and LIDAR, camera images capture the characteristics of a materials’ texture rather than directly making distance measurements. Therefore, when using image-based localization we need to exploit other geometric information to obtain 3D poses. However, these approaches are prone to error when there is perspective variation as well as change in illumination. While this challenge has been heavily studied such as the works of McManus et al. [2014] and Porav et al. [2018], it remains a major limitation for operation in all weather and lighting conditions.

In contrast, other sensors have been designed specifically to work in such conditions such as Frequency-Modulated Continuous Wave RADAR (FMCW). Due to its wavelength, FMCW RADARs can propagate through rain and dust and also operate at long ranges. The drawback of this type of sensor is that the data it provides is not as informative as images nor as interpretable as LIDAR. RADAR localization remains a research topic where recently quite promising results have been achieved that might enable robust localization in the near future. The works of Park et al. [2019b], Park et al. [2019a], Aldera et al. [2019], and Cen and Newman [2019] provide detailed description of RADAR localization.

In between, LIDAR is a sensor that measures 3D information directly and provides very reliable measurements. Despite its high cost, it is the preferred sensor modality for applications such as obstacle avoidance and odometry. Shan and Englot [2018] demonstrated exceptional motion estimation results based on LIDAR sensing. One of the drawbacks of raw LIDAR measurements is that individual points are uninformative in isolation. A set of points lacks the descriptive texture when compared to a set of RGB pixels. This motivated the

exploration of LIDAR-based approaches which leverage a larger collection of point information (segments) in a point cloud.

Other sensor options include GPS and SONAR. Of course, GPS can give very precise position estimates, especially RTK-GPS. [Levinson et al. \[2007\]](#) demonstrated that GPS could be used as a prior for localization in urban environments. However, GPS does not provide an orientation measurement, while also being heavily degraded or absent in forests and indoor environments. The physical transduction of SONAR sensors is similar to RADAR — transmitting relatively long wavelengths through water enables measurements at much longer ranges underwater than passive cameras. In the marine domain, [Hover et al. \[2012\]](#) and [Fallon et al. \[2013\]](#) showed that SONAR is a key technology especially in turbid water. However, in air it is noisy, short range, and indistinct. While it has been used for room-level localization by [Liu et al. \[2017\]](#), it is not widely studied.

Finally, the most mature approaches use a combination of several sensor inputs to achieve the most reliable localization [[Thrun et al. 1999](#); [Levinson et al. 2007](#); [Dissanayake et al. 2001](#)].

2.2.2 LIDAR localization in Robotics

LIDAR localization has been studied since the 1990's [[Thrun et al. 1999](#)]. It remains an essential approach for many robotics applications due to the reliability of LIDAR sensors to produce accurate 3D point clouds. The next paragraphs present an overview of LIDAR localization methods most relevant to this thesis.

[Wolcott and Eustice \[2017\]](#) used LIDAR reflectivity to perform localization against road marking to reliably localize a vehicle in an urban environment. The authors modelled the world as a mixture of Gaussians (GMM) over 2D grid cells. The GMM represented the heights of points in each cell and the reflectance in a manner which was robust to weather alteration and road degradation. However, the authors run an exhaustive search over the entire GMM space, resulting in a $O(n^3)$ complexity.

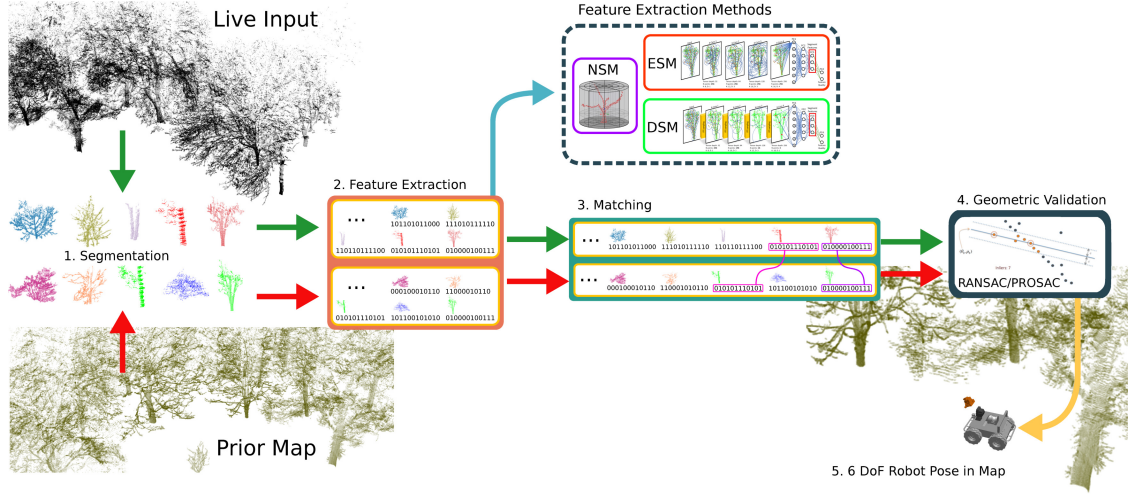


Figure 2.3: Segmentation-based approach to global localization.

Baldwin and Newman [2012] fused two 2D LIDARs, a vertical push-broom and horizontal sensor, into a 3D swathe. The fusion of the LIDAR measurements into a swathe was done based on the motion of a vehicle. The swathes were localized with respect to a previously acquired map. The approach used Gaussian Processes to model the underlying velocities in a probabilistic framework. For each 3D swathe, the objective function minimized the Kullback-Leibler divergence of the distribution of projected 3D points against the prior map. Unfortunately, the approach was not robust to highly dynamic environments, as the push-broom sensor scanned the environment only once, and integrated the measurements over time.

Hess et al. [2016] leveraged the robot’s odometry as a rough estimate of the position of the robot within the world. In order to compute loop closures in real-time, the authors used a branch-and-bound method for computing scan-to-submap matches as constraints. This kept the computational costs low by avoiding searching for loop closures over the entire map.

2.2.3 Segment-based LIDAR Localization in Robotics

A recent approach for global LIDAR localization is to extract small point cloud clusters or *segments* from the raw input point clouds which are then matched against a prior map and used for localization (fig. 2.3). A segment is a cluster

of LIDAR points broadly but not directly comparable to an object. It might correspond to a patch of a flat wall, a vehicle, or a tree trunk. Each extracted segment represents to a key pose relative to the sensor when they were extracted. Therefore, the segment-based localization approach can be reformulated as recognizing individual segments in the current scene, that are also present in the same relative geometric position in the prior map. These segments represent landmarks in the environment that are unique, repeatable, and easily matchable. This approach is used instead of determining local keypoints or directly registering the raw points individually (as in the case of ICP).

This approach was first proposed by Douillard et al. [2012] where segments constituted the middle ground between local and global approaches. Global approaches can suffer as they are not robust to occlusions or change in the environment, but they can encapsulate the context of the landmarks/objects present in the scene. Conversely, local methods, similar to the works of Zhuang et al. [2012] and Steder et al. [2011], were designed to be robust to occlusions and change, but because they only characterized local regions they lost much of the contextual information. Segments, however, are large enough that they contain reliable and repeatable features while also being robust to occlusions and temporal changes.

Segment localization were first applied to LIDAR localization in the works of Dubé et al. [2017], Dubé et al. [2018], and Dubé et al. [2019] and this thesis draws a lot of inspiration from these works. The SegMatch approach extracted more reliable feature matches than the then state-of-the-art 3D point descriptors and could achieve a large number of robust localizations. Similarly to Zach et al. [2015], SegMatch’s localizations are obtained by a geometric match pruning of the proposed segments. The centroids of a set of segments are used as input to RANSAC [Fischler and Bolles 1981] which produces the final pose estimate. To create the list of proposed segments as an input to RANSAC, different approaches rely on various machine learning methods. Cramariuc et al. [2018] compared three different feature extraction strategies for segment

description based on 3D Convolutional Neural Networks (CNN) - simple classifier approach, Siamese-type of network, and a descriptor network trained with contrastive learning. In SegMap, segment localization evolved into a segment-based SLAM that introduced a new segment feature that also encoded semantics and volumetric shape [Dubé et al. 2018]. This novel approach learned features from a voxelized input space that were suitable for localization and semantic description of the map. The approach was evaluated in a realistic urban setting and in industrial buildings. Elbaz et al. [2017] used segments produced from a Random Sphere Cover Set - overlapping point cloud spheres where each point of the original point cloud could be part of more than one sphere. These spheres were then projected to 2D depth images and processed by a deep auto-encoder. In their approach when segmenting the point clouds into spheres, the authors assumed that each super-point described a surface.

2.2.4 Segment-based localization problem statement

In the following paragraphs, we will formally define the main approach to global localization. The definition is a crucial step towards understanding the particularities of the approach.

There are two established ways of performing localization:

1. coarse methods to solve the global alignment problem,
2. ICP and its variants to carry out fine-registration

The former methods, known as loop closure detectors, aim to find a correspondence between a current scene and a previously acquired map. An advantage of global localization algorithms is the absence of an initial guess of where the sensor is located. Therefore, the primary application for these approaches is when the localization is performed in an enormous map, perhaps consisting of hundreds of kilometres. Fine registration methods estimate an alignment based on the LIDAR point correspondences between consecutive scans. Their alignment is very precise, however, they need an initial pose estimate and often

can fall into local minima. These approaches are frequently used to perform odometry estimation based on the successive point cloud data.

The goal of global registration algorithms is to localize within a prior (target) map, P^T , using the current live (source) point cloud, P^S . The output is a transformation, defined as ${}^T T_S$ that minimizes the distance $d(p_s, p_t)$, $\forall p_s \in P^S$ and $\forall p_t \in P^T$. It accurately computes the location of the sensor in a prior map. The transform is a 6 Degrees of Freedom (DoF) transformation, ${}^T T_S \in \text{SE}(3)$.

Features are low-dimensional compact representations of raw data that include summary information about a set of point or a full point cloud. It is normal for a target map to consist of hundreds of millions of data points, justifying the use of features to represent and register point clouds. Usually features are described in a fixed-length descriptor vector that can be compared to a different vector of the same length to compute a measure of similarity. 3D features describe a specific point in space or position, using geometric information of the surrounding points. The global alignment problem can therefore be reformulated as extracting robust and repeatable features in the current scene, that are also present in the prior map. These features are then described with a feature descriptor that allows us to match similar landmarks in P^T and P^S point clouds. Finally, an optimization is run in order to compute the best transformation which aligns the feature correspondences.

A feature descriptor can be categorized as being local or global. Local descriptors take individual points as input ($\forall p \in P^{T,S}$) and describe the local geometry around that point. Therefore, the feature vector, $F(p_i)$, is computed only on specific points in a scene that contains geometric information. These points are defined as keypoints, p_i for $i \in P^{T,S}$ [Bosse and Zlot 2013; Zhuang et al. 2012; Steder et al. 2010a; Steder et al. 2011; Gawel et al. 2016]. In contrast, global descriptors describe the whole point cloud with a single descriptor [Röhling et al. 2015; Granström et al. 2011; Magnusson et al. 2009; Angelina Uy and Hee Lee 2018]. These approaches often try to match a scene point cloud to a database

storing previous observations of point clouds, thus recognizing where the vehicle was located. Other approaches focus on a collection of 3D points as a place of interest, known as segments. They usually require segmentation as a preprocessing step to extract relevant clusters of points [Dubé et al. 2017; Dubé et al. 2018]. These approaches split the environment into a collection of segments and store the map segments in a database, while matching the currently observed segments to the database. In such a way, these algorithms are robust to occlusions in very dynamic environments. For segment-based approaches, one needs to define a *keypoint* that represents the segment. These keypoints are stored in the database, instead of storing the whole segment, which is much more efficient. The *keypoints* are processed during the geometric verification stage. Commonly, the mean of the entire segment is used as a keypoint [Dubé et al. 2017].

Once the feature description of the point clouds has been computed, $F(P^T)$ and $F(P^S)$, the next step is to compare the descriptor vectors for a scene point cloud to the database of descriptors using K-nearest-neighbor (KNN) search in feature space. This results in a set of correspondences $C = \{c_i\}$, where $c_i = \{p_i^T, p_i^S\}$. Here p_i^T and p_i^S refer to the keypoint i in the target map and source point clouds respectively. Finally, RANSAC [Fischler and Bolles 1981] is initialized in 3D with the set of correspondences C , which computes a geometrically consistent 6 DoF pose. This will provide a coarse alignment of the local scene within the prior map. The computed 6 DoF pose estimate can also be used as an initial guess to ICP for a fine refinement of the pose [Elbaz et al. 2017].

2.3 Feature learning from 3D data

The principal weakness of traditional global localization approaches that use LIDAR data is their reliance on hand-crafted features, due to the difficulties in learning from unstructured point clouds. For image-based localization, learned features now outperform hand-crafted features [Simo-Serra et al. 2015]. How one would achieve similar improvements with LIDAR data was not initially

evident, as continuous 3D data is not naturally supported by such learning networks. Recently works by [Dubé et al. 2018] and our own contribution, chapter 4, have however shown that learned LIDAR features can now out perform hand-crafted features.

The main consideration when learning from LIDAR data is how to represent it. There are two common options: using a voxelized spatial representation or defining the 3D data as an unstructured set of points. The following section presents different methods suitable for learning from LIDAR data.

2.3.1 Voxel-based approaches

The voxelized representation was first used because convolution filters could be more easily adapted to voxel space, because a voxel space is a regular 3D grid which is but a higher dimensional representation of 2D pixel space. Voxelized approaches such as [Choy et al. 2016] achieved promising results but had to cope with the immense computational requirements of performing convolutions on a voxel space. A $32 \times 32 \times 32$ voxelized grid was used, resulting in a dimensionality of 32,768, with 3D convolutions applied in strides. This was quite a significant computational requirement, while at the same time only being able to have a very coarse representation of the 3D space. Due to the input resolution having to be so low (due to the computational complexity), such approaches were not able to handle detailed shapes such as the legs of a chair, for example [Haoqiang et al. 2017].

Li [2017] proposed 3D-FCN for vehicle detection in LIDAR data. It operated by using 3D convolutions on top of raw LIDAR data to produce an objectiveness map and a bounding box for each object prediction. The former classified whether a region belongs to an object, while the latter estimated a bounding box of this object. It used a combination of softmax loss for the objectiveness map and an L2 loss between the predicted bounding box and the ground truth annotated one.

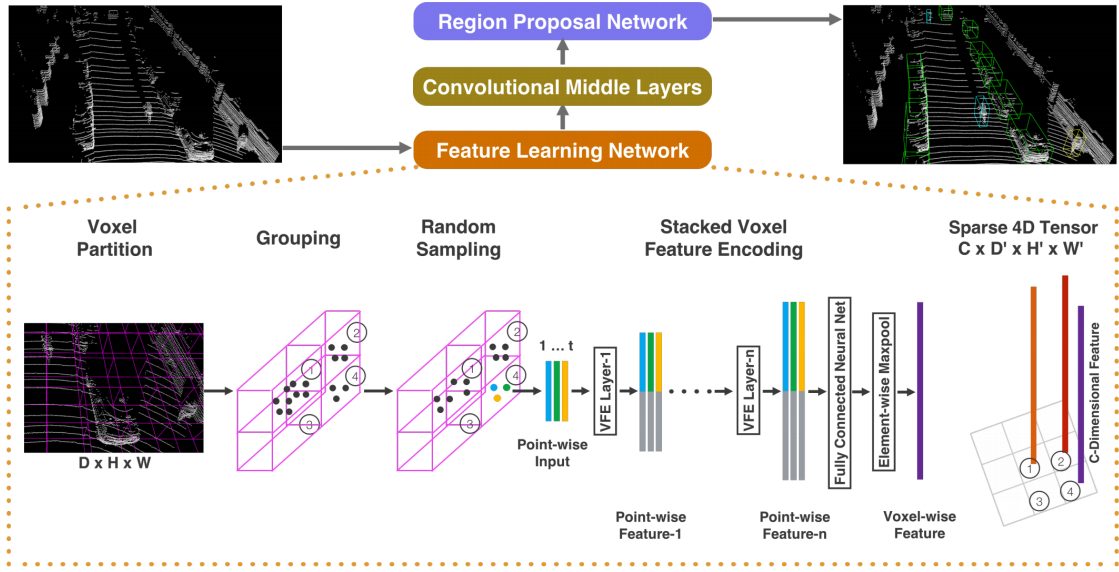


Figure 2.4: Example of voxel-based network architecture. Image adopted from [Zhou and Tuzel 2018].

VoxelNet, depicted on fig. 2.4, extended 3D-FCN by incorporating point-wise features, instead of raw point values in the 3D CNN architecture [Zhou and Tuzel 2018]. The authors proposed a Voxel Feature Encoder layer to efficiently compress the sparse 3D points to a list of local features. The features were further aggregated via 3D convolutions and fed to a Regional Proposal Network (RPN) [Ren et al. 2015] for fast object detection.

Engelcke et al. [2017] proposed a feature-centric voting scheme for sparse point cloud processing named Vote3Deep. By incorporating the novel layer and combining it with L1 regularization, Vote3Deep encouraged sparsity in the intermediate layers of the architecture. This allowed for shallow network architectures to outperform deeper models on object classification tasks. This is useful due to the computation constraints that deeper networks impose.

2.3.2 Point-based approaches

Later, directly using an unstructured set of points as input was popularized by PointNet [Qi et al. 2017a; Qi et al. 2017b]. By learning a symmetry function approximated by a multi-layer perceptron (MLP) the authors proposed a new kind of layer that could learn an approximation of convolutional operator over

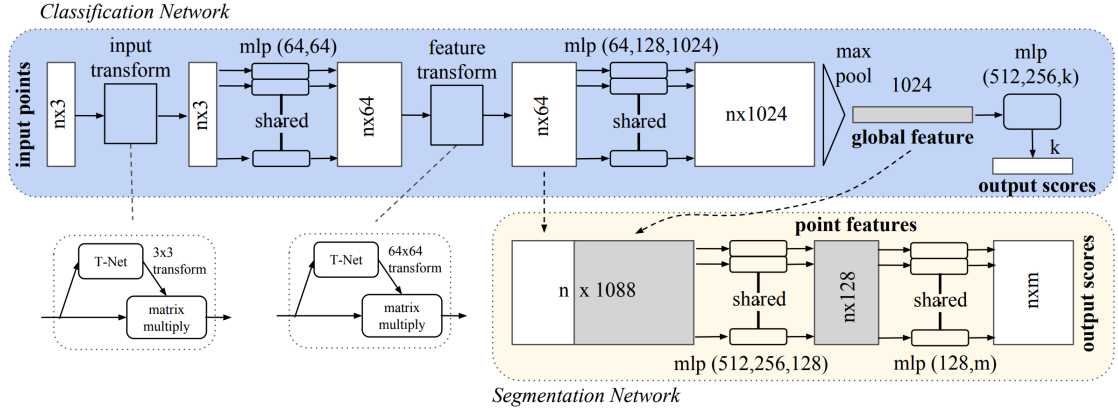


Figure 2.5: Example of point-based network architecture. Image adopted from [Qi et al. 2017a].

sets of unordered points (fig. 2.5). In contrast to the aforementioned volumetric approach, a raw processed point cloud containing 1,024 points resulted in a dimensionality of only 3,072. In a similar fashion, Li et al. [2018] managed to increase the performance of the basic neurons used to handle unordered point clouds by also approximating a convolution using a MLP and by performing aggregation of spatial data within each neuron.

Angelina Uy and Hee Lee [2018] presented a place recognition method where a point cloud was down-sampled and matched against other point clouds from a map to perform place recognition. The point clouds were described using a combination of networks trained with a lazy quadruplet loss that produced a 256-dimensional feature vector.

Qi et al. [2018] utilized PointNet for object detection in 3D data. Their approach, named Frustrum PointNets, first generated a 2D image proposal in RGB space using a CNN. Then, the corresponding 3D points inside the projected frustrum were processed by PointNet in the depth dimension. Finally, the object predictions were generated by performing 3D instance segmentation and amodal 3D bounding box prediction for the points inside the frustrum.

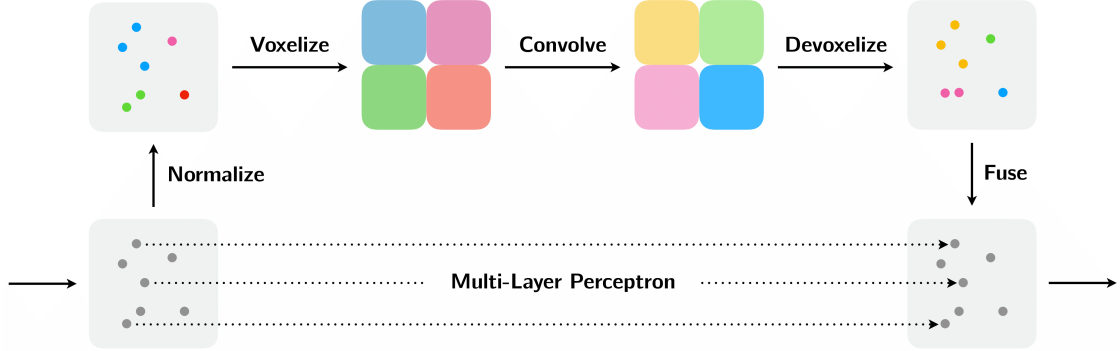


Figure 2.6: Example architecture that fused point-based input with voxel-based convolutions. Image adopted from [Liu et al. 2019].

2.3.3 Other 3D learning approaches

There also exist a number of approaches that do not directly learn from raw point cloud data, but use other point cloud representations. Some utilize the geometry of the scene or represent the point cloud as a graphs and leverage its properties. In the following we outline a few examples of these types of approaches.

Dynamic Graph CNN (DGCNN) introduced a novel convolutional operator named EdgeConv [Wang et al. 2019]. EdgeConv operated on dynamically generated graphs at each layer of the architecture. Instead of treating each point individually, it combined local neighborhood information and considered the geometry around each point. It was a widely used alternative to PointNet-type architectures for object part segmentation, object detection, and classification.

Li et al. [2019] introduced auxiliary neural network operations to graph convolutional neural networks. They demonstrated that deep graph CNNs can be more efficient than their shallow counterparts. By adapting residual/dense connections, dilated convolutions, and dropout to graph-based CNN approaches, they demonstrated that these concepts directly translate from images and improve the state-of-the-art for both PointNet and DGCNN networks.

PVCNN combined the best of both worlds in terms of accuracy and computational performance. Liu et al. [2019] benefited from both the memory improvements of using points as input, and the reduced sparse data access that comes from voxel-based convolutions (fig. 2.6). PVCNN consisted of a *high-resolution*

point-based branch and a *low-resolution* voxel-based branch. The voxel-based branch extracted coarse neighborhood information, which was fused by the fine-grained features of the point-based branch. In this manner Liu et al. [2019] were able to produce state-of-the-art results at a fraction of the computational cost.

2.4 Keypoint extraction

Feature-based SLAM methods extract and describe key areas in the environment, for example landmarks, obstacles or any other keypoints [Cadena et al. 2016]. This can be seen as an alternative to the segmentation in segment-based localization approaches, where we explicitly define a set of points in the environment and consider the region around each keypoint as a *segment*. This could be based either on a fixed radius around the keypoint or on K-nearest neighboring (KNN) points around the keypoint. Since chapter 6 focuses on reliable extraction of keypoints, in the following we review classic approaches to extract keypoints in both images and LIDAR data.

2.4.1 Extracting keypoints in images

Extracting keypoints in images has grown to be rotation and scale invariant, and has gained a lot of popularity in the last 20 years. We split this section in classic, hand-engineered methods for extracting keypoints, and more recent approaches that learn to extract keypoints from data.

Classic image keypoint extraction approaches. Harris corner detection is one of the most widely used approaches to detect keypoints in images [Harris and Stephens 1988]. It detected large variation in intensity in all directions of a certain image patch. Non-maximum suppression was applied as the last step in order to pick optimal values that represent corners. Figure 2.7a visually presents the difference of intensities in terms of eigenvalue variations of an image patch with a corner region. The corner detection algorithm was invariant to rotations, but not to scale variations.

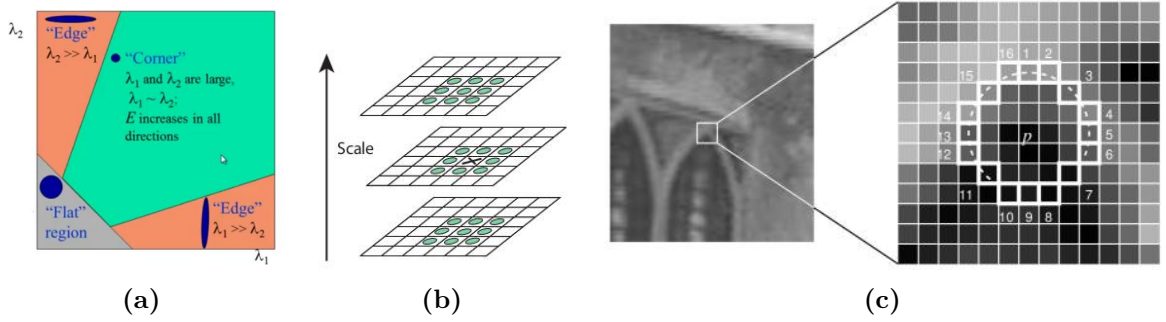


Figure 2.7: Keypoint extraction methods. a) Harris corner extraction [Harris and Stephens 1988], b) SIFT local maxima keypoint extraction in different scales [Lowe 2004], c) FAST keypoint extraction for a keypoint p and a circular neighborhood of pixels 1-16 [Rosten and Drummond 2006]. Image adopted from corresponding works.

Lowe [2004] proposed an alternative that was both rotation and scale invariant, named SIFT. It uses Difference of Gaussians at different scales of the image in a Gaussian-blurred pyramid. Once the differences are computed, the local maxima over space and octaves were found. Figure 2.7b shows an example of an extracted keypoint at three different scales and 9 different spatial locations. Contrast thresholding was applied to remove outliers such as low-contrast keypoints and edge points. Finally, an orientation was assigned to each keypoints based on the gradient of each keypoint and its surroundings and scale.

Rosten and Drummond [2006] introduced an alternative to SIFT, named FAST, that was also able to run in real time. Their approach took an image pixel p and its intensity I_p and considered the 16 neighbors in a circle around p , as illustrated on fig. 2.7c. If p was a corner, there existed a set of N contiguous pixels that were either brighter or darker than I_p . There was also a high speed test, that only considers neighbors 1, 5, 9, and 13 and if all of them are brighter or darker, the full test is performed. The limitations of FAST are that some non-corner pixels were not rejected and some were considered multiple times. These were all addressed with a random forest classifier and a non-maximum suppression.

Learned image keypoint extractors. Recently there has been a lot of traction about how to learn usable keypoints from images due to the rise of the deep

learning era. Below we outline some of the more recent methods for keypoint extraction in images.

Verdie et al. [2015] are one of the pioneers in this field by learning a keypoint detector named TILDE. It used an existing detector, SIFT, to generate initial keypoints candidates that are not repeated in 100 images. Then they generated saliency maps from these keypoints and learned interesting pixels via supervision. A major limitation of their detector was the need for external detector as part of their training procedure.

Yi et al. [2016] introduced LIFT, a method for jointly learning keypoint description and detection. They used 3 CNNs - one for learning saliency maps, and the other two for description, based on a local patch extracted around the keypoint. The extracted keypoints were the maxima of the saliency map for each image. The two description CNNs were trained first in order to initiate the training, for which corresponding matching pixels are required. This *warm-up* procedure is necessary in order for the detector to learn based on the descriptor.

Very recently Benbihi et al. [2019] proposed an approach where no learning is needed, but instead the detector relied on a pretrained descriptor network. Therefore, they did not need any supervision, nor fine-tuning for their algorithm, making it an attractive choice for keypoint extraction. They showed state-of-the-art results based on popular learning architectures such as VGGNet [Simonyan and Zisserman 2014], AlexNet [Krizhevsky et al. 2012], Xception [Chollet 2017]. Examples of keypoints extracted from the saliency maps of VGGNet are shown in fig. 2.8. Chapter 6 draws inspiration from this approach and applies it to LIDAR data, and also proposes a further learning procedure to improve results.

2.4.2 Extracting keypoints from 3D data

Many of the keypoint extraction ideas on images are directly translatable to 3D space. However, one could also consider additional features of 3D points such as curvature, normals, or intensity information. In the following we detail engineered and learned methods for keypoint extraction in RGB-D and LIDAR data.

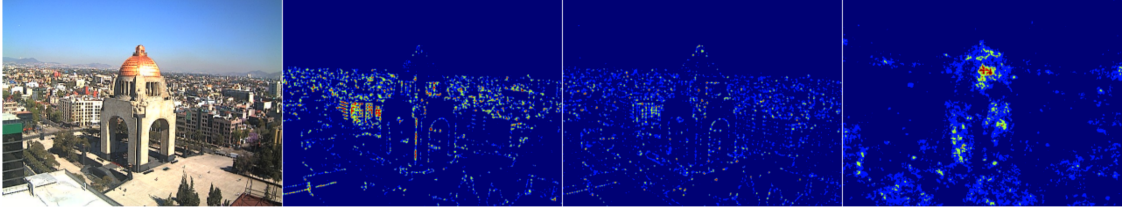


Figure 2.8: Learned keypoints Left to right: Image from Webcam dataset, saliency maps activations for layers *pool 3*, *pool 2*, and *pool 5* of VGGNet [Simonyan and Zisserman 2014]. Image adopted from [Benbihi et al. 2019].

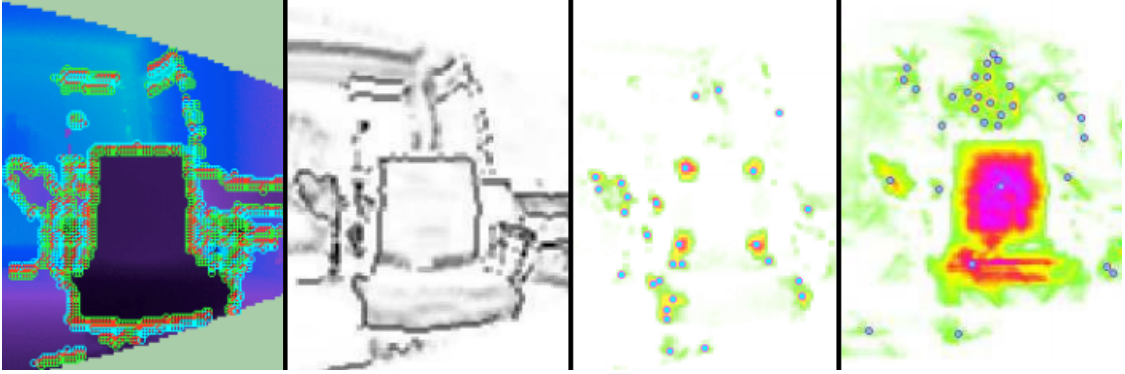


Figure 2.9: Keypoint extraction example in 3D data. Left to right: range image of a scene with a chair, the extracted curvature, corresponding keypoints extracted at 0.2m and 1.0m support size. Image adopted from [Steder et al. 2010b].

Engineered approaches to keypoint extraction in 3D data. There have been several attempts to extend SIFT to three dimensions [Ni et al. 2008; Scovanner et al. 2007; Flitton et al. 2010; Rister et al. 2017], however, all of these approaches considered medical image analysis as their primary domain of operation. The basic SIFT keypoint detector can be applied to point clouds by first approximating a density function to sample the data regularly. Second, the scale space was built based on the density function by a number of Gaussian kernels applied to the input point cloud. Third, a local maxima was computed in the resulting scale space. Finally, the keypoints were positioned based on the scale space extremes of the Difference of Gaussians [Azimi et al. 2019].

Steder et al. [2010b] proposed the Normal Aligned Radial Feature (NARF) keypoint extractor for range images (fig. 2.9). The point cloud was first converted to a range image representation. Second, a surface change score was computed based on the principal curvature and borders of objects in the scene.

Keypoints were then extracted based on the surface score stability in a given region and the difference in dominant direction. Finally, smoothing and non-maximum suppression in three dimensions were applied to filter out local interest points. One limitation of NARF is that it still required to transform the raw data to range images for processing.

Intrinsic Shape Signatures (ISS) [Zhong 2009] is a 3D keypoint extraction algorithm that performed Principal Component Analysis (PCA) on a weighted covariance matrix in order to create an intrinsic reference frame F_i . The two dominant eigenvalues were used as the x and y axis, with their cross product as the z axis around a basis point p_i . F_i is view-point invariant, as all four possible intrinsic reference frames were considered (performing PCA on the structure tensor results in orientation ambiguity around each axis of 180°). Zhong [2009] utilized a spherical coordinate system around p_i , and converts the sphere to an octahedron, which is further discretized into a spherical grid. This resulted in histogram bins with invariant size and noise invariant around the centre origin p_i . ISS was adopted by Holz et al. [2015] and Gawel et al. [2017] as a 3D keypoint detection method.

Learning keypoint extraction from 3D data. Next, we present state-of-the-art methods for learned keypoint extraction directly from data.

PointSIFT was the first attempt to bring SIFT to point-type based networks [Jiang et al. 2018]. Since CNNs were scale and translation invariant by design, the authors introduced an orientation-encoding point convolutional layer to make the framework rotation invariant as well. The authors encoded eight orientations and provided a multi-scale representations by stacking several orientation-encoder units. While they did not solve the keypoint detection problem explicitly, they focused directly on the final end-task goal. Their encoder-decoder type of architecture achieved state-of-the-art results on semantic segmentation tasks.

3DFeatNet was the first approach to consider joint learning of keypoints and a feature descriptor on 3D LIDAR data. Yew and Lee [2018] proposed an alignment and attention-based mechanism to detect keypoints. The authors

used a PointNet architecture with a triplet of positive, negative, and anchor point clouds that were processed with a Siamese type of network and learned via a triplet loss. The detector learned a weight per point of how useful each interest region is, while the descriptor took a cluster of points around each interest point for description. Their approach was weakly supervised by only requiring tagged point clouds by a GPS/INS sensor, however, the descriptor had to be pretrained for a couple of epochs, before the detector can learn any useful keypoints.

Li and Lee [2019] extended 3DFeatNet to work in an unsupervised scenario, where an example point clouds was randomly rotated and this transformation was given to the network a-priori with both rotated and original clouds. Their model, named USIP, learned based on a probabilistic Chamfer loss and point-to-point loss which aided high repeatability and accurate keypoint localization. Their approach was evaluated for both matchability and repeatability on the ModelNet40 dataset and achieved superior results to other hand-crafted and learned alternatives. One limitation of USIP is the introduced bias of the network, by reusing the same data points as input.

2.5 Summary

This chapter reviewed the concepts and relevant literature of LIDAR odometry and localization in the context of simultaneous localization and mapping. Furthermore, it provided a thorough understanding of the problem of segment-based global localization. Segment-based localization methods are expressive and also robust to occlusions, which presents a good medium between approaches based on local registration and global descriptors. Segment-based methods present interesting research directions. For example, one can reconstruct the segments on a *viewer* computer in order to efficiently visualize the output of a SLAM system while keeping the network bandwidth low. It is, however, an open research question how to carry out coarse registration given the segment association.

Currently, the majority of methods rely on RANSAC or clique-based techniques to estimate a coarse alignment, which does not always have consistent solutions.

With the recent developments in deep learning it is now possible to process 3D LIDAR data, however, we still require large GPU clusters to train these methods efficiently. Moreover, large scale annotated datasets are needed for these types of algorithms to learn. This requires expensive supervision that is often intractable to obtain. A recent trend in deep learning is to create approaches that require weak or no supervision.

Keypoint extraction not only follows the tendency of requiring less supervision, but recent approaches are even claiming to work without any training whatsoever. This is an interesting concept to explore, as it suggests that pre-trained networks already possess the required information to extract interesting regions in a scene.

3

Seeing the Wood for the Trees: Reliable Localization in Urban and Natural Environments

The main goal of this chapter is to evaluate localization systems on different types of environment. Our key contributions involve designing a hybrid feature selection strategy that allows a place recognition system to work better than other state-of-the-art methods in challenging outdoor environments. The features descriptor comprised of a combination of geometric features and features describing the distribution of 3D points around a keypoint. We also propose a new keypoint selection strategy for each individual object that is to be matched between a current scene of the environment and a database. In this chapter we present an approach, named Natural Segmentation and Matching (NSM) for localization in urban, parkland, and forested environments.

The manuscript was presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2018 [Tinchev et al. 2018].

Seeing the Wood for the Trees: Reliable Localization in Urban and Natural Environments

Georgi Tinchev, Simona Nobili and Maurice Fallon

Abstract—In this work we introduce Natural Segmentation and Matching (NSM), an algorithm for reliable localization, using laser, in both urban and natural environments. Current state-of-the-art global approaches do not generalize well to structure-poor vegetated areas such as forests or orchards. In these environments clutter and perceptual aliasing prevents repeatable extraction of distinctive landmarks between different test runs. In natural forests, tree trunks are not distinctive, foliage intertwines and there is a complete lack of planar structure. In this paper we propose a method for place recognition which uses a more involved feature extraction process which is better suited to this type of environment. First, a feature extraction module segments stable and reliable object-sized segments from a point cloud despite the presence of heavy clutter or tree foliage. Second, repeatable oriented key poses are extracted and matched with a reliable shape descriptor using a Random Forest to estimate the current sensor’s position within the target map. We present qualitative and quantitative evaluation on three datasets from different environments - the KITTI benchmark, a parkland scene and a foliage-heavy forest. The experiments show how our approach can achieve place recognition in woodlands while also outperforming current state-of-the-art approaches in urban scenarios without specific tuning.

I. INTRODUCTION

Localization is an important problem in autonomous robot navigation when surveying challenging environments such as forests, oil rigs, and disaster response sites or teach-and-repeat operations in orchards for disease detection and analysis of tree growth. Systems should be able to recognize places in both urban and vegetated areas, while being resilient to appearance changes caused by the robot’s motion, occluding clutter or temporal variations of the environment.

There has been extensive study of localization in indoor or outdoor structured environments using vision or laser sensors. Strategies are often split into two parts - global registration approaches which match high level features within a large map [1], [2], [3], and fine registration methods that perform point-wise refinements [4], [5]. These strategies often rely on landmark correspondences between candidate point clouds, implicitly assume built environments, containing planar objects, or require reliable normal estimation and consistent data association.

The authors are with the Oxford Robotics Institute at the University of Oxford, United Kingdom. {gtinchev, snobili, mfallon}@robots.ox.ac.uk

S. Nobili is with the Institute of Perception, Action and Behavior, School of Informatics, University of Edinburgh, UK.

This work was supported by the EPSRC RAIN and ORCA Robotics Hubs (EP/R026084/1 and EP/R026173/1 respectively). M. Fallon is supported by a Royal Society University Research Fellowship.

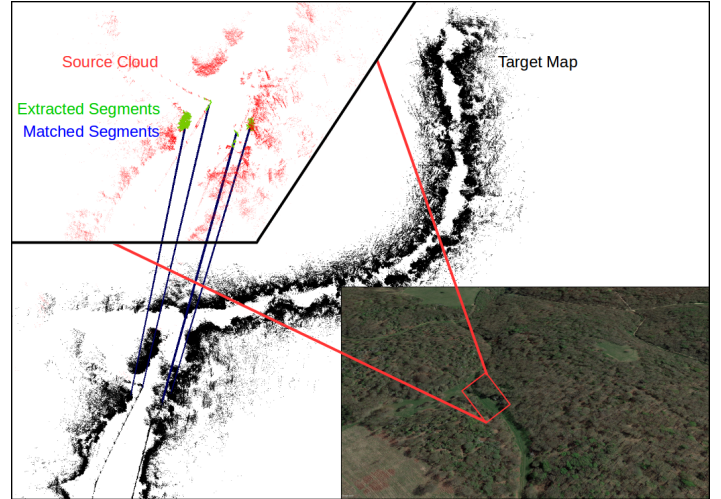


Fig. 1: The proposed approach provides reliable segmentation (green) and matching (blue lines) of objects such as trees and bushes within natural, forested environments. The current *source* point cloud (red) is shown in the top left corner. The blue lines indicate the localization estimate relative to the *target* map (black), consisting of a previously traversed ~ 1 km route within a forest.

The problem of place recognition in structure-poor scenarios is less well explored. Current approaches model natural environments specifically, using state machines to aid their place recognition systems based on prior knowledge about the dataset [6], [7]. It is common to specifically model the canopy of trees or bushes. The foliage in vegetated areas changes with vegetation growth or pruning, and is also affected by continuous seasonal alteration. In addition, defining an accurate localization is difficult because the tree canopy can be interleaved and heavy foliage/clutter can obscure stable landmarks.

It is currently unclear how localization approaches will transfer to the more challenging environments we are interested in. Designing algorithms to be invariant to the type of environment and geometrical differences may result in performance improvement. In this work we bridge the gap between structured urban environments and challenging natural environments without explicitly modeling every orchard or forest individually by using more repeatable and descriptive features.

The main contribution of our work is a method that enables global place recognition¹ in three different scenery settings. Our approach adapts the framework from [2] to reliably extract segments from natural environments which carry repeatable and distinctive information from a *source* point cloud and utilize them to solve global registration to a *target*

¹In this system no initial estimate of the sensor’s pose is needed to initialize the alignment.

map (Fig. 1). The pipeline of our approach is illustrated in Fig. 2.

Our contributions are as follows:

- 1) we propose an extension of the SegMatch approach [2] for place recognition in both urban and natural environments, which we call Natural Segmentation and Matching (NSM),
- 2) we identify and evaluate a novel combination of key pose extraction and description methods. The key pose extraction module segments and defines consistent orientated coordinate frames for object-sized segments despite the presence of heavy clutter. The descriptor carries sufficient information to recognize different instances of the same segment, achieving a higher accuracy with respect to previous approaches,
- 3) we perform a thorough evaluation of the approach across datasets captured in an urban area, a parkland scene and a foliage-heavy forest.

The remaining sections of the paper are structured as follows: Sec. II presents the literature review, Sec. III outlines each module of our algorithm, and Sec. IV extensively evaluates the approach. We discuss interesting findings and limitations of our approach in Sec. V.

II. RELATED WORK

Global localization is commonly solved by directly extracting and describing keypoints from a point cloud [1], [8], [9], or by segmenting objects in the scene and matching those to a prior map [2], [3]. A coarse alignment is computed that can later be refined with methods like the Iterative Closest Point (ICP) [10]. The task is more challenging in natural environments, as keypoint extraction or segmentation methods often assume some regular structure in the environment.

Place recognition strategies

Bosse and Zlot [8] presented three different methods for keypoint selection that were described with model grid descriptors. The descriptors' dimensions were later normalized using a nonlinear normalization function and reduced to increase efficiency and reduce the signal-to-noise ratio. Place recognition was achieved by keypoints *voting* for their closest neighbor in a previous database of keypoints. Their work was extended in [9] to support 3D regional point descriptors.

Elbaz et al. [3] utilized a Random Sphere Cover Set (RSCS) to divide a point cloud into a set of super-points clusters. Each point can be a part of multiple super-points, resulting in the point cloud being separated by overlapping spheres. These spheres were projected as depth map images and fed into a Deep Auto-Encoder network to generate feature descriptors. Candidate matches are selected using a K-nearest neighbors (k-NN) search. In their approach when segmenting the point clouds into RSCS, the authors assumed that each super-point described a surface.

Dubé et al. [2] presented SegMatch. This approach segmented the target and source point clouds using Euclidean clustering. These clusters were matched using k-NN and

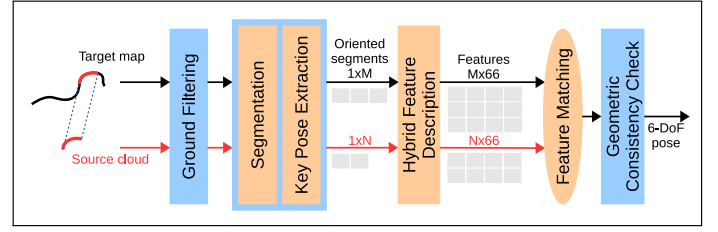


Fig. 2: The proposed approach.

predictions from a Random Forest, trained on an Eigenvalue-based and Ensemble of Shape Histogram features. The centroids of the matched segments were then used to produce a 6-DoF pose estimate for the sensing vehicle using RANSAC. The approach assumes that the centroid (mean) of all points is a representative point of a cluster, which unfortunately does not hold in natural environments.

Localization in Natural Environments

Some of the challenges in natural environments include occlusions, clutter or branch deformations, unstable normal extraction, predominance of structure-poor objects. These are usually the result of interleaved trees or even just windy conditions. Thus most state-of-the-art approaches performing place recognition in natural environments rely on explicit assumptions about the model of the scene [6], [7].

Wellington et al. [6] presented an approach for tree segmentation in orchards, utilizing a push-broom laser scanner. The ground surface was estimated using a Markov Random Field, which helped estimating the boundaries between each tree, the height and density of trees. A Hidden Semi-Markov Model modeled the environment using three states - tree, gap or boundary. The approach incorporated a hidden state to account for the explicit prior on tree spacing. Underwood et al. [7] extended this approach to perform tree recognition and platform localization. After segmentation and characterization, the localization problem was modeled as a Hidden Markov Model [11]. The approach was tested in an orchard with regular rows of trees which were clearly delineated.

Bosse and Zlot [1] performed place recognition in natural environments by selecting a set of keypoints from 10% of the densest areas in a point cloud, and computed a local Gestalt descriptor around each of them [8]. The method was evaluated on a natural open eucalyptus forest with multi-use dirt trails, where the densest areas of the point cloud were vegetated. In the proposed work, we utilize a different strategy for keypoint extraction which is based on firstly extracting segments in the environment, which we then append with a common orientation frame and describe using a Gestalt descriptor. A notable difference is that we opted for a learning-based approach when matching between key poses.

In summary, our work differs from the state-of-the-art in place recognition in natural environments in that we utilize segmentation as a prior step for key frame extraction on natural data. Our work adapts SegMatch [2] to reliably segment key poses in manner which is repeatable between different observations. In addition, we utilize a feature de-

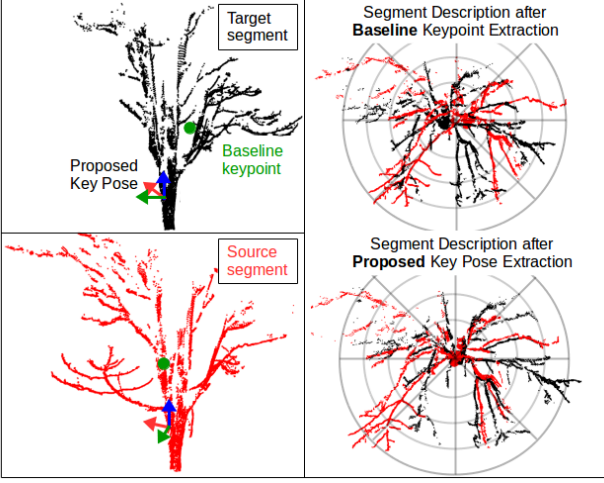


Fig. 3: Key pose estimation between two different observations of the same segment: the target map (black) and source cloud (red). The centroid of each segment shown in green, used by a baseline method [2]. The proposed method computes position and orientation for the key pose which is consistent for both the target and source clouds. The figures on the right represent a top view of the segments aligned with either of these two approaches. Subsequently our key pose descriptor can more coherently extract features as the points of each segment occupy the same descriptor bin in the two observations.

scriptor that describes both the shape of the segment and the corresponding key pose’s neighborhood, resulting in resilient localization in cluttered natural areas and urban scenarios.

III. NATURAL SEGMENTATION AND MATCHING

This section presents the proposed Natural Segmentation and Matching (NSM) approach, which adapts SegMatch [2] for natural environments. We wish to estimate the position of the sensor within a target map using the current source point cloud. The approach focuses on identifying repeatable segments from point clouds captured in cluttered scenes where the detected objects can differ significantly in appearance between observations. We aim to overcome the weaknesses of the previous methods by improving the feature extraction and description steps as follows: A) prior to segmentation, the input clouds are pre-filtered using a Progressive Morphological Filter to discard ground points [12]. Then oriented key poses are computed for each segmented object. This makes the extraction more robust to variation in appearance from different points of view. B) A hybrid feature descriptor is defined using PCA and Gestalt features, which embed sufficient information so as to enable place recognition in both natural and urban environments.

A. Feature Extraction

Segmentation: We employ Euclidean segmentation, similarly to [2], in order to delineate individual objects such as interleaved trees and bushes. A typical segment corresponds to a tree with a portion of its major branches, a dense bush, a vehicle, some urban structure or any rigid object in the scene with limited physical size. Most clutter is implicitly filtered-out as it does not satisfy criteria about the uniformity of points and dimensions of the segment in question. Care is taken to ensure that it is possible to reliably distinguish a specific object across temporal or spatial variations by assigning a key pose to each segment.

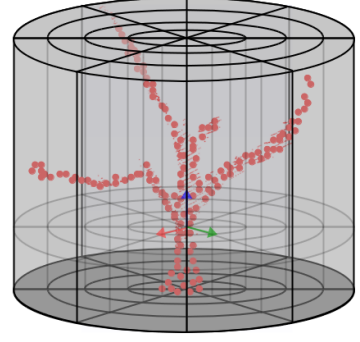


Fig. 4: Visual representation of the Gestalt descriptor of a tree segment (red), oriented around the extracted key pose.

Note, that we considered the region-growing approach of [13], however, normals computed in these natural environments are unstable [1].

Oriented Key Pose Extraction: For each segment in the source cloud $s_{si} \in \{s_{s1}, s_{s2} \dots s_{sN}\}$, and in the target map $s_{tj} \in \{s_{t1}, s_{t2} \dots s_{tM}\}$, a key pose $\mathbf{k} \in \mathbb{R}^6$ needs to be defined. This pose will be used to define the position and orientation of each segment so as to aid the recognition task. An illustration of how a consistent key pose can help mitigate changes in appearance between different points of view is shown in Fig. 3.

We assume that the observed objects are rigidly connected to the ground. This allows us to orient the z -axis of $\mathbf{k}$ as the global up vector, and determine the orientation of the segment based on the predominant dimension along which the points are distributed perpendicularly to z . The x -axis is projected onto the normal direction at $\mathbf{k}$, and the y -axis is computed as the cross product between the x and z axes. Once the orientation is been determined, the key pose’s position $[k_x, k_y, k_z]$ is calculated as the median of all the points of a subset of s and stored for later use. The median is used instead of the mean so as to mitigate segmentation differences between matching objects as shown on Fig. 3

B. Hybrid Feature Description

In the next step, the extracted segments are described in both the target and source clouds using a hybrid feature set combining *Eigenvalue* features and an adaptation of the *Gestalt* model [1], [14], illustrated in Fig. 4. Thus, NSM encodes both the geometric properties of the segments and their point distribution.

For each segment s , planarity and cylindricity features are computed after PCA. These are, respectively,

$$P_\lambda = \lambda_2 - \lambda_1 \quad (1)$$

and

$$C_\lambda = \lambda_3 - \lambda_2, \quad (2)$$

where $\lambda_{\{1,2,3\}}$ are to the ordered normalized eigenvalues. The structure tensor is of rank 3 and it follows that $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. P_λ and C_λ encode the geometric information for each cluster, enabling NSM to account for the rigidity of objects.

Each segment is then split into 4 radial and 8 azimuthal divisions about its key pose's z -axis and clock-wise starting from the x -axis, resulting in 32 bins. For each bin the mean and variance of the height values of all points are recorded for a total of 64 dimensions. The planarity and cylindricity values are included in the feature vector, resulting in a descriptor of 66 dimensions. In the experimental section we show that such a descriptor carries sufficient information to identify different instances of the same segment in cluttered scenarios.

C. Feature Matching

Feature matching is performed in two stages using the approach proposed by [2]. Firstly, a k -NN search in feature space creates a list of possible matches for each source segment s_{s_i} that matches K target segments using L_2 distance. Secondly, the list of candidate matches is further refined using a Random Forest (RF) that performs a binary classification over each set of K proposals.

We have trained our RF using the full feature vector dimension while taking into consideration the difference between two matching samples. The input to our RF consists of $(|\mathbf{f}_s|, |\mathbf{f}_t|, \Delta\mathbf{f})$, where $\mathbf{f}_s$ and $\mathbf{f}_t$ are the feature vectors corresponding to all source segments and its candidate matches in the target map and $|\cdot|$ denotes the modulus. We consider the descriptor $\Delta\mathbf{f}$ to be computed from the difference between each source and target feature vector, that is $\Delta\mathbf{f}_{ik} = |\mathbf{f}_{s_i} - \mathbf{f}_{t_k}| \forall i, k$ with $k \in [1 : K]$, which we empirically found to help during the matching task. This is possible as the segments have a consistent orientation after key pose extraction and each feature vector is sorted by definition. The output of the RF classifier is a score w , defining the likelihood of a match being correct between segments s_s and s_t . The score is thresholded to produce a set of accepted matches.

D. Pose Estimation

We utilize the method from [18], [2] in order to maintain geometric consistency between each pair of matches $((\mathbf{k}_{s_p}, \mathbf{k}_{t_p}), (\mathbf{k}_{s_q}, \mathbf{k}_{t_q})) \in \mathcal{C} = \{(\mathbf{k}_{s_0}, \mathbf{k}_{t_0}) \dots (\mathbf{k}_{s_L}, \mathbf{k}_{t_L})\}$, where $\mathcal{C}$ is the set of accepted matches proposed by the RF and L its cardinality. The formula in Eq. (3) ensures that

Parameters		KITTI	GS	CP
Segmentation	Min # Points	200		2500
	Max # Points	1500		50000
	Max Distance		0.2m	
Description	Gestalt Radius		2.0m	
Matching	K Neighbors		200	
	# of Trees		250	
	Tree Depth		50	
	RF Threshold (w)		0.69	
Pose Estimation	Resolution (ϵ)		0.4m	
	Min # Clusters (τ)		4	

TABLE II: Parameters used in our experiments.

a pair of matching key poses have similar spatial distances between them.

$$\| \mathbf{k}_{t_p} - \mathbf{k}_{t_q} \|_2 - \| \mathbf{k}_{s_p} - \mathbf{k}_{s_q} \|_2 < \epsilon \quad (3)$$

In Eq. (3), ϵ is the resolution parameter, which determines how similar the source cloud is to the target. The lower the value, the stricter the algorithm behaves. A localization proposal is accepted only if it satisfies a criteria about the minimum number of accepted matches. For a 6-DoF pose to be estimated, there must be at least $\tau = 3$ matches. Finally the 6-DoF transform which aligns the source cloud into the target map is computed by a RANSAC optimization on the key poses of the remaining matches.

IV. EXPERIMENTAL EVALUATION

The main goal of this work is to perform global place recognition relative to a prior LIDAR map, without any prior information about the sensor's current pose. Our experiments are designed to demonstrate the capabilities of our system and are performed on three different datasets with increasing complexity, as shown on Tab. I. Their corresponding parameters are presented in Tab. II.

The RF used in all of our experiments was trained on Seq. 06 of the KITTI dataset, using a careful manual annotation of matching pairs of segments between individual clouds. We identified 487 true matches (distance between segments $< 0.5m$) and 218513 negative segment correspondences (distance $\geq 0.5m$) in this manner.

	KITTI	George Square Park (GS)	Cornbury Park (CP)
Source	Geiger et al. [15]	Ours	Ours
Environment	Urban, structured	Vegetated, structured, well observed trees	Vegetated, unstructured, hard to delineate trees, predominance of bushes
Dynamics	Moving cars, bicycles, pedestrians	A few moving people	None
LIDAR	3D HDL-64E @ 10Hz	Push-broom @ 75Hz	3D HDL-32E @ 10Hz
Path	On streets	3 loops on a concrete trail in the park	Off-road dirt track out and back trail
Scene Area	$\sim 500m$ for localization	$\sim 500m$ each loop	$\sim 1km$ each direction
# of Scans	1104	~ 40700	1823
# of Points	$\sim 130000/\text{scan}$	$\sim 381/\text{scan}$	$\sim 70000/\text{scan}$
Map creation	Scans projected on ground truth	VO [16] on first loop, manual loop closures [17]	Scans projected on ground truth from forward pass locally corrected via ICP globally corrected via [17]
Ground Truth	✓	✗	✓
Training Data	KITTI Seq 06	Pre-trained KITTI	Pre-trained KITTI
Testing Data	KITTI Seq 00	Georges Square (GS)	Cornbury Park (CP)
Experiment	Exp. A	Exp. B	Exp. C

TABLE I: Datasets used in our experiments.



Fig. 5: The robots utilized to collect data for our experiments. A Clearpath Husky robot (left) was used to collect the GS dataset with a push-broom LIDAR and a Multisense SL. A Bowler Wildcat platform (right) was utilized to collect the CP dataset, equipped with a Velodyne HDL-32E LIDAR, Oxts GPS, Novatel INS and Bumblebee 2 camera.

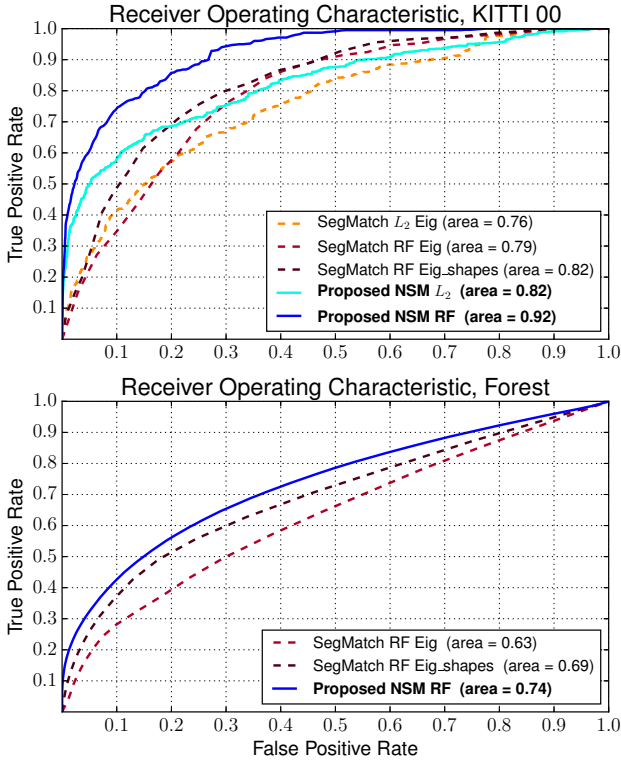


Fig. 6: Receiver Operator Characteristic curves for NSM and SegMatch [2] evaluated on Seq 00 of the KITTI dataset (top) and a forest dataset, similar to GS (bottom). The proposed approach was evaluated with both k-NN in feature space (L_2 , cyan) and a pre-trained Random Forest (RF) classifiers (blue). The baseline was evaluated with Eigenvalue-based (Eig) features and Eigenvalue with ESH (Eig_shapes) features.

The proposed approach was tested against SegMatch [2] in all experiments, which is indicated as a baseline where appropriate. Both NSM and SegMatch produce localization estimates relative to a prior map, for every 1 meter of vehicle distance traveled.

To evaluate the accuracy, the estimated pose T_e , was compared to the best ground truth pose we could achieve, T_c . The error ΔT is computed as follows:

$$\Delta T = \begin{bmatrix} \Delta \mathbf{R} & \Delta \mathbf{t} \\ 0 & 1 \end{bmatrix} = T_e T_c^{-1} \quad (4)$$

where $\Delta \mathbf{t}$ represents the translation error and $\Delta \mathbf{R}$ the rotation error in matrix form. The 3D translation error e_t is defined as the Euclidean distance of the translation vector $\Delta \mathbf{t}$ as follows:

$$e_t = \|\Delta \mathbf{t}\| = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \quad (5)$$

The 3D rotation error e_r is defined as the Geodesic distance given the rotation error $\Delta \mathbf{R}$ as follows:

$$e_r = \arccos\left(\frac{\text{trace}(\Delta \mathbf{R}) - 1}{2}\right) \quad (6)$$

The choice of evaluation metric is motivated by [4].

The following three experiments were carried out to support our claims:

- A) Quantitative analysis of the performance of the proposed hybrid feature descriptor on urban and forested data with comparison to the baseline.

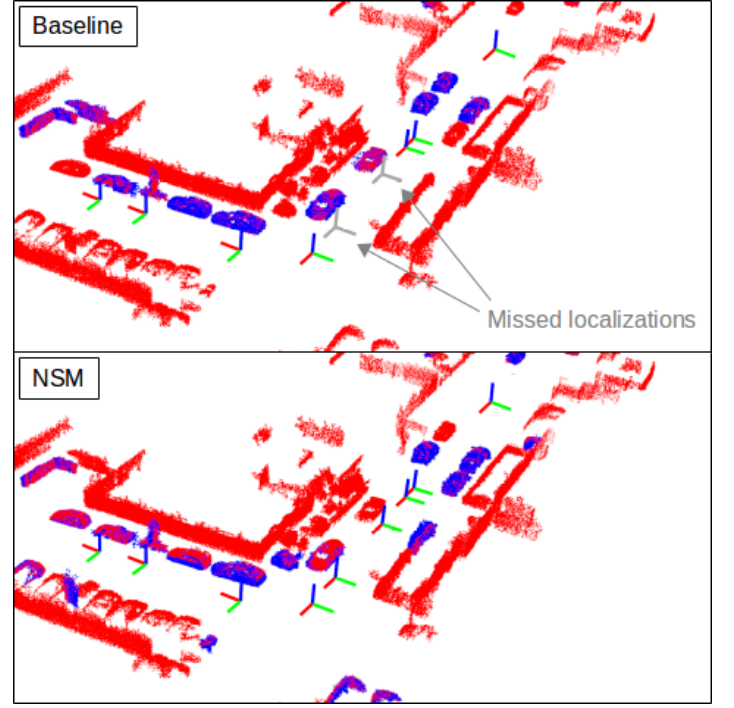


Fig. 7: Qualitative illustration of the performance of our algorithm (bottom) on the KITTI dataset in comparison to the baseline (top). Blue point clouds represent correctly identified segments in the source cloud (red). Axis-colored poses indicate successful localizations, while gray colored are missed localizations.

- B) Both qualitative and quantitative evaluation of our method performing in parkland environment, showing that the extracted segments are better represented by the oriented key pose resulting in more accurate localizations.
- C) A demonstration of the performance of our localization system in a very heavily vegetated environment.

A video to accompany this paper is available at <http://ori.ox.ac.uk/nsm-localization>.

A. Feature Description Analysis

The first experiment examines the performance of our hybrid feature descriptor. We use an approach similar to that presented in [2]. We use the KITTI dataset and a forested environment similar to the GS dataset. The KITTI dataset was captured by a 3D Velodyne HDL-64E LIDAR mounted on a car in an urban environment. The forested dataset was recorded with a 3D Velodyne VLP-16 mounted on top of a Clearpath Robotics Husky UGV, as shown in Fig. 5.

Fig. 6 (top) illustrates the Receiver Operating Characteristic (ROC) curves for three combinations of features and two classifiers. Namely, we compared Eigenvalue-based features and Ensemble of Shape Histogram (ESH) features, as proposed in [2] to our hybrid feature extractor. Furthermore, we compared the performance of a RF classifier trained on the aforementioned features to the L_2 distance in feature space (k-NN). Our proposed approach extracted more positive matches than the baseline with both L_2 distance in feature space (cyan in comparison to dashed orange) and a RF classifier (blue compared to dashed brown), while limiting false positive correspondences. In the illustrative example

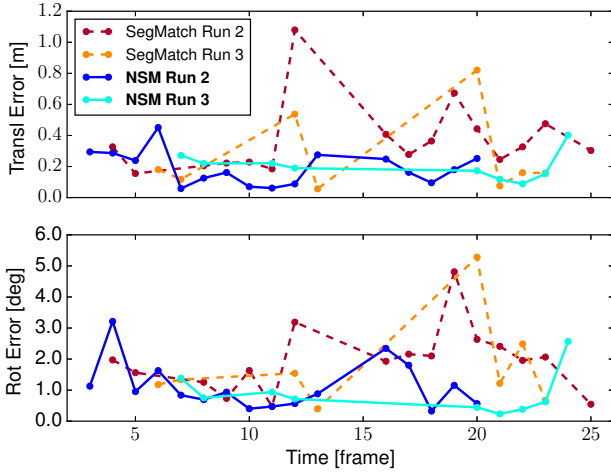


Fig. 8: Translation (top) and rotation (bottom) errors for the proposed approach (continuous lines) and the baseline algorithm (dashed lines) tested on the GS dataset from two sequential runs. Run 1 was used to create the map.

		Transl RMSE	Rot RMSE	Localizations	Frames
SegMatch	Run 2	$0.43\text{m} \pm 0.22\text{m}$	$2.22^\circ \pm 1.03^\circ$	16	26
	Run 3	$0.37\text{m} \pm 0.25\text{m}$	$2.28^\circ \pm 1.45^\circ$	8	29
NSM	Run 2	$0.22\text{m} \pm 0.11\text{m}$	$1.35^\circ \pm 0.76^\circ$	16	26
	Run 3	$0.22\text{m} \pm 0.09\text{m}$	$1.12^\circ \pm 0.67^\circ$	9	29

TABLE III: Quantitative results of the proposed approach and the baseline on GS dataset.

on Fig. 7, we show that the proposed approach was able to produce two more localizations as it successfully matched partial observations of segments from the front on the vehicle (bottom of Fig. 7).

Fig. 6 (bottom) illustrates the ROC curve of the proposed features’ classifier against the baseline on the forest dataset. The performance of the all the classifiers is significantly lower than on the urban dataset. The proposed feature extraction method has larger area under the curve (AUC) compared to the baseline. This result could be improved by learning features, optimized for specific environments. We plan to explore this topic in our future work.

Based on these results, we chose a false positive rate of 0.1 (RF score $w = 0.69$) for our subsequent experiments in order to limit false positive segment matches and maximize performance.

B. Performance in a Parkland Environment

In a second experiment we aim to showcase the proposed key pose selection strategy of our system in a park environment. It consists of well-observed trees, benches and bushes spatially separated and with few interleaved branches between the trees.

This experiment was carried out in George Square Park, Edinburgh, with a Clearpath Husky robot (Fig. 5, left) traversing three loops of $\sim 500\text{m}$ each. The first loop was used to create the map (Run 1), while loops two and three (Run 2, 3) were used to test the localization. The vehicle was equipped with a push-broom LIDAR. Visual odometry [16] was used to create 3D LIDAR swathes for both the prior map and the source clouds. The prior map was further corrected in a SLAM system using [17] by carefully identifying

loop closures. For the source cloud, LIDAR swathes were accumulated every $\sim 10\text{m}$ that the vehicle traveled. To account for the different sensor classes the segmentation algorithm parameters were changed as listed in Tab. II. In this experiment we compared NSM to the baseline using centroid computation, Eigenvalue features and the RF classifier. Note that we tried utilizing Eigenvalue+ESH features with a pre-trained RF, however, no localization proposals were produced with this system configuration.

Fig. 8 illustrates translation and rotation error as computed using Eq. (4) from the last two test runs. A visual representation of the environment and a running example of it are shown on Fig. 9 (left). Neither the baseline, nor the proposed approach produced any false localizations. Quantitative results of the experiment are shown in Tab. III. The proposed approach shows better accuracy for localization. We speculate that in the case of NSM the oriented key pose extraction strategy contributes for a closer match to the ground truth estimate, as previously illustrated on Fig. 3.

C. Performance in a Foliage-Heavy Forest

The last experiment demonstrates the potential of our system to localize in highly vegetated scenes. The data for the experiment was collected in Cornbury Park, Oxfordshire, while traversing an approximately 1km path through woodland. The data was captured by a Bowler Wildcat equipped with a Velodyne HDL-32E LIDAR, producing 3D point cloud scans at 10Hz (Fig. 5, right). Pictures from a camera mounted on a robot in this environment are shown in Fig. 9 (right). This environment was very challenging to localize in, even for a person manually comparing the prior map to a source point cloud. It consists of heavily interleaved foliage, trees and bushes without clear spatial separation.

Because the vehicle was equipped with a combined GPS/INS sensor, we intended to use this for evaluation. However, the route along the path combined periods of GPS reception and periods where trees covered the path and blocked GPS reception. Instead, we created the map by combining an initial vehicle trajectory, made by registering scans spaced every 30m to one another using ICP, with carefully chosen and reliable GPS measurements. To do this we smoothed the initial trajectory with the GPS measurements using [17] to achieve a best estimate of ground truth. Source point clouds were created when traversing the environment in the opposite direction by accumulating point clouds for every $\sim 1\text{m}$ traveled.

Fig. 10 presents the translation and rotation errors of correct and incorrect localization matches, computed using Eq. (4). In the environment we discovered three sections of uninterrupted bushes obscuring any trees (marked with 1, 2, 3 on Fig. 10 and Fig. 9, right). In these areas we did not expect the algorithm to function, as the continuous hedge was visually challenging even for a person to distinguish. Outside these areas there were 96.7% positive and 3.32% false localization matches. We envisage that the two false detections were due to the radius of the Gestalt feature, as this feature’s descriptiveness depends on the size and

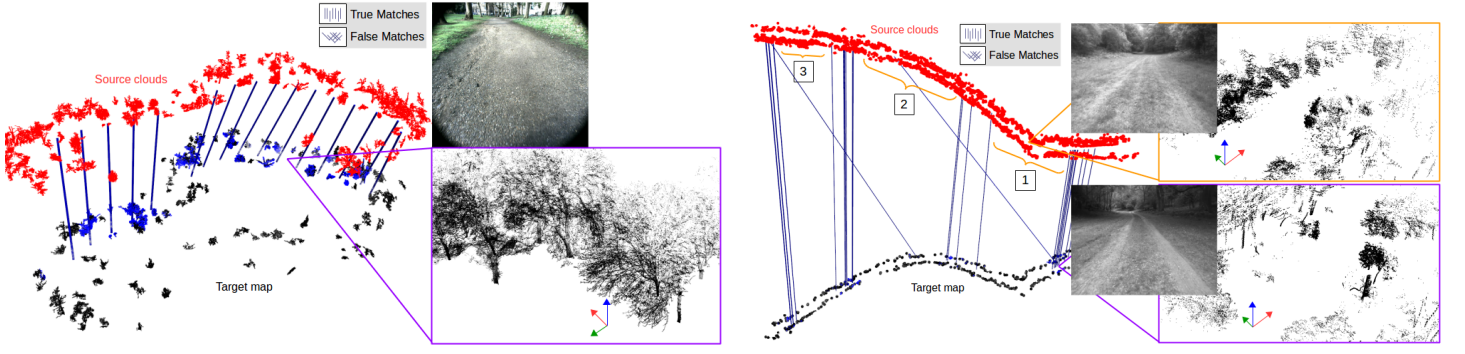


Fig. 9: Illustration of the proposed approach on the GS dataset (left) and CP dataset (right). In both environments the source clouds (red) are being registered to an *a-priori* target map (black). An example of a point cloud during successful localization in the presence of trees is shown in purple, while an example of a missed localization (observed in 1, 2, 3) in the absence of rigid objects is illustrated in orange. Images from the robot's point of view are shown to illustrate the challenging scenario.

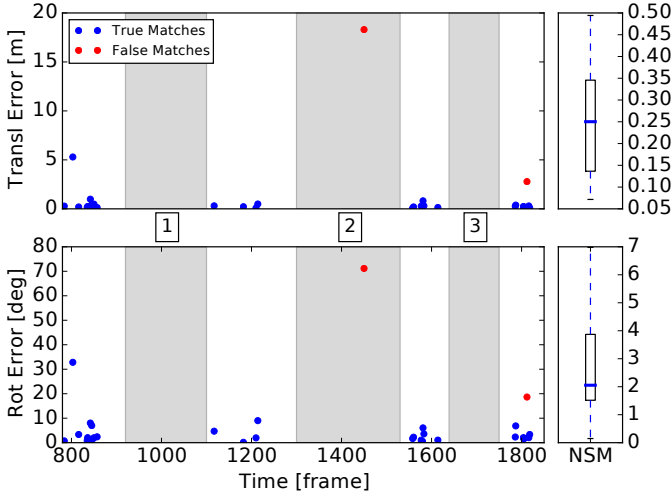


Fig. 10: 3D translation error (top) and rotation error (bottom) of correct and incorrect localizations, evaluated on the CP dataset. True matches (blue) correspond to the consistent vertical lines in Fig. 9 (right), while incorrect localizations (red), correspond to inconsistent diagonal lines in Fig. 9 (right). Box plots on the right indicate the quantiles for the distribution of 0.50 (thick blue bars), 0.25 and 0.75 (lower and upper black rectangles) of true localization matches. During the experiment the vehicle traversed areas of continuous vegetation (1, 2, 3).

symmetry of the objects. In those specific cases two small bushes, with boundary points of their convex hull equal to the Gestalt radius, were incorrectly matched to bigger bushes that fully encapsulated them.

In this environment the descriptiveness of the features enabled the proposed approach to produce a series of successful localizations. This was achieved by identifying partially observed and asymmetrical segments.

V. DISCUSSION AND LIMITATIONS

As described above the proposed approach is able to provide reliable localization proposals in urban and natural environments without specific parameter tuning, despite encountering challenging vegetated scenes. The current limitations of the algorithm are:

- 1) The segmentation of large objects can differ between the target map and the current point cloud when the sensor vantage point changes significantly. This would result in dissimilar segments which could not be matched.

- 2) The Gestalt descriptor has bins of varying sizes and the descriptor's azimuthal divisions intersect near the key pose. Thus, the feature descriptor is vulnerable to the precise location of the key pose.
- 3) There are no constraints on the location of the segments with respect to the vehicle, which influences the way RANSAC estimates a registration. In forests and orchards the simple regularity can cause the step to struggle such as in frame 803 of Exp. C (Fig. 10), where the error is noticeably higher than the RMSE. This is still considered as a true localization (blue dot $\approx 5m$ RMSE and 30° rotation error), as the segments are matched correctly but the precise alignment was incorrect due to all the segments being located on one side of the vehicle. This will be addressed in our future work.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented Natural Segmentation and Matching, a novel approach for feature description and matching for global localization in natural and urban environments. Our method exploits rigid objects which are repeatable and distinctive in unstructured scenes which allows us to successfully recognize previously visited places. We implemented our approach and evaluated it on an urban, park, and heavily vegetated datasets. The experiments demonstrated that NSM performs comparably to state-of-the-art approaches in an urban environment, while also outperforming these algorithms in natural environments without specific parameter tuning.

In future plans we aim to extend the algorithm to learn environment specific features and to perform multi-seasonal evaluation.

VII. ACKNOWLEDGMENT

We would like to thank the authors of SegMatch [2] for open sourcing their implementation and sharing valuable insights about it.

REFERENCES

- [1] Michael Bosse and Robert Zlot. Place recognition using keypoint voting in large 3D LIDAR datasets. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2677–2684. IEEE, 2013.

- [2] Renaud Dubé, Daniel Dugas, Elena Stumm, Juan Nieto, Roland Siegwart, and Cesar C. Lerma. SegMatch: Segment based place recognition in 3D point clouds. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5266–5272, 2017.
- [3] Gil Elbaz, Tamar Avraham, and Anath Fischer. 3D Point Cloud Registration for Localization using a Deep Neural Network Auto-Encoder. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 2472–2481, 2017.
- [4] François Pomerleau, Francis Colas, Roland Siegwart, and Stéphane Magnenat. Comparing ICP variants on real-world data sets. *Autonomous Robots*, 34(3):133–148, April 2013.
- [5] Jacopo Serafin and Giorgio Grisetti. NICE: Dense normal based point cloud registration. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 742–749, Sept 2015.
- [6] Carl Wellington, Joan Campoy, Lav Khot, and Reza Ehsani. Orchard tree modeling for advanced sprayer control and automatic tree inventory. In *IEEE/Intl. Conf. on Computer Vision Workshops (ICCV Workshops)*, pages 5–6, 2012.
- [7] James P Underwood, Gustav Jagbrant, Juan I Nieto, and Salah Sukkarieh. LIDAR-based tree recognition and platform localization in orchards. *J. of Field Robotics*, 32(8):1056–1074, 2015.
- [8] Michael Bosse and Robert Zlot. Keypoint design and evaluation for place recognition in 2D LIDAR maps. *Journal of Robotics and Autonomous Systems*, 57(12):1211–1224, 2009.
- [9] Michael Bosse and Robert Zlot. Place recognition using regional point descriptors for 3D mapping. In *Field and Service Robotics*, pages 195–204. Springer, 2010.
- [10] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Machine Intell.*, 14(2):239–256, February 1992.
- [11] Lawrence R Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–286, 1989.
- [12] Keqi Zhang, Shu-Ching Chen, Dean Whitman, Mei-Ling Shyu, Jianhua Yan, and Chengcui Zhang. A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE Transactions on Geoscience and Remote Sensing*, 41(4):872–882, 2003.
- [13] Tahir Rabbani, Frank Van Den Heuvel, and George Vosselmann. Segmentation of point clouds using smoothness constraint. *ISPRS Annals of the Phot., Remote Sens. and Spatial Inf. Sciences*, 36(5):248–253, 2006.
- [14] Alex Walthelm. Enhancing global pose estimation with laser range scans using local techniques. In *Intl. Conf. on Intelligent Autonomous Systems (ICIAS)*, pages 1120–1130, 2004.
- [15] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *Intl. J. of Robotics Research*, 32(11):1231–1237, 2013.
- [16] Albert S. Huang, Abraham Bachrach, Peter Henry, Michael Krainin, Daniel Maturana, Dieter Fox, and Nicholas Roy. Visual Odometry and Mapping for Autonomous Flight Using an RGB-D camera. In *Intl. J. of Robotics Research*, pages 235–252. Springer, 2017.
- [17] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Trans. Robotics*, 24(6):1365–1378, 2008.
- [18] Aitor Aldoma, Federico Tombari, Luigi Di Stefano, and Markus Vincze. A global hypotheses verification method for 3D object recognition. *Eur. Conf. on Computer Vision (ECCV)*, pages 511–524, 2012.

Summary

In this chapter we evaluated state-of-the-art localization systems on both urban and natural environments. By following the main topic of the thesis we implemented and evaluated a localization system robust to both environments.

One of the main observations of is that hand-engineered features failed to generalize between natural and urban scenarios. More specifically, shape geometry information alone is not enough to capture the details of the natural scene. Therefore, we proposed more complex feature descriptor that considers the distribution of points around a specific key point. Our next observation is that the definition of these keypoints is not straightforward and can vary significantly, especially in natural environments, where the object is rarely uniformly captured in two consecutive observations.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Seeing the Wood for the Trees: Reliable Localization in Urban and Natural Environments
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Tinchev, G., Nobili, S., & Fallon, M. (2018, October). Seeing the wood for the trees: Reliable localization in urban and natural environments. In <i>2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)</i> (pp. 8239-8246). IEEE.

Student Confirmation

Student Name:	Georgi Tinchev		
Contribution to the Paper	- Primary author of the work - Lead the development of the data collection - Collected the data - Developed a modular framework for LIDAR loop closure detection - Designed experiments - Performed analysis on different types of features and environment - Experimentally evaluated different classifiers suitable for segment matching - Collaborated on writing the manuscript		
	Date	30/12/2020	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Dr. Maurice Fallon		
Supervisor comments: The contributions listed above are accurate		
	Date	30/12/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

4

Learning to See the Wood for the Trees: Deep Laser Localization in Urban and Natural Environments on a CPU

Motivated by the findings in chapter 3, we turn to deep learning approaches suitable for learning features from 3D data. Our main objective is to push the boundaries of existing hand engineering features used for place recognition in challenging environments. Another key goal is to create a compact system that does not require heavy computational resources during operation. In this chapter, we present an improvement of NSM from chapter 3, named Efficient Segment Matching (ESM) — an approach suitable for real-time operation in a variety of challenging environments.

This work was published in the *IEEE Robotics and Automation Letters*, 2019. The manuscript was presented at the *IEEE International Conference on Robotics and Automation*, 2019 [Tincev et al. 2019].

Learning to See the Wood for the Trees: Deep Laser Localization in Urban and Natural Environments on a CPU

Georgi Tinchev, Adrian Penate-Sanchez and Maurice Fallon

Abstract—Localization in challenging, natural environments such as forests or woodlands is an important capability for many applications from guiding a robot navigating along a forest trail to monitoring vegetation growth with handheld sensors. In this work we explore laser-based localization in both urban and natural environments, which is suitable for online applications. We propose a deep learning approach capable of learning meaningful descriptors directly from 3D point clouds by comparing triplets (anchor, positive and negative examples). The approach learns a feature space representation for a set of segmented point clouds that are matched between a current and previous observations. Our learning method is tailored towards loop closure detection resulting in a small model which can be deployed using only a CPU. The proposed learning method would allow the full pipeline to run on robots with limited computational payload such as drones, quadrupeds or UGVs.

I. INTRODUCTION

Localization is a fundamental task in robotic perception, a robot needs to know where it is to navigate in the environment and to make decisions. It has been heavily explored with computer vision, demonstrating impressive results at large scales [1], [2], [3]. These types of approaches typically assume a certain inherent structure in the scene, image features are dependent on repeatable camera viewpoint [3], and methods are often tested in urban environments which guide the robot along the route in question [4]. While the aforementioned visual approaches have many promising characteristics, here we explore LIDAR due to its robustness to varying lighting conditions, changes in viewpoint, and track line offsets. It is a precise and long range sensing modality.

The SegMatch system [5] proposed a modular segment-based approach for LIDAR teach-and-repeat, which could localize within a prior map while also retaining a degree of semantic meaning, as the segments matched corresponded to large physical objects such as cars and parts of a building. The approach was further improved with learned feature descriptors [6] which achieved greater accuracy. The approach uses a GPU to achieve real-time performance.

We are motivated to localize in natural scenarios like forests. In these environments many assumptions about the appearance or the geometry of landmarks are no longer valid - vegetation grows between seasons and there is no

The authors are with the Oxford Robotics Institute, University of Oxford, United Kingdom. {gtinchev, adrian, mfallon}@robots.ox.ac.uk

This work was supported by EPSRC RAIN and ORCA Robotics Hubs (EP/R026084/1 and EP/R026173/1 respectively) and EU H2020 project Memory of Motion (MEMMO, project ID: 780684). M. Fallon is supported by a Royal Society University Research Fellowship.

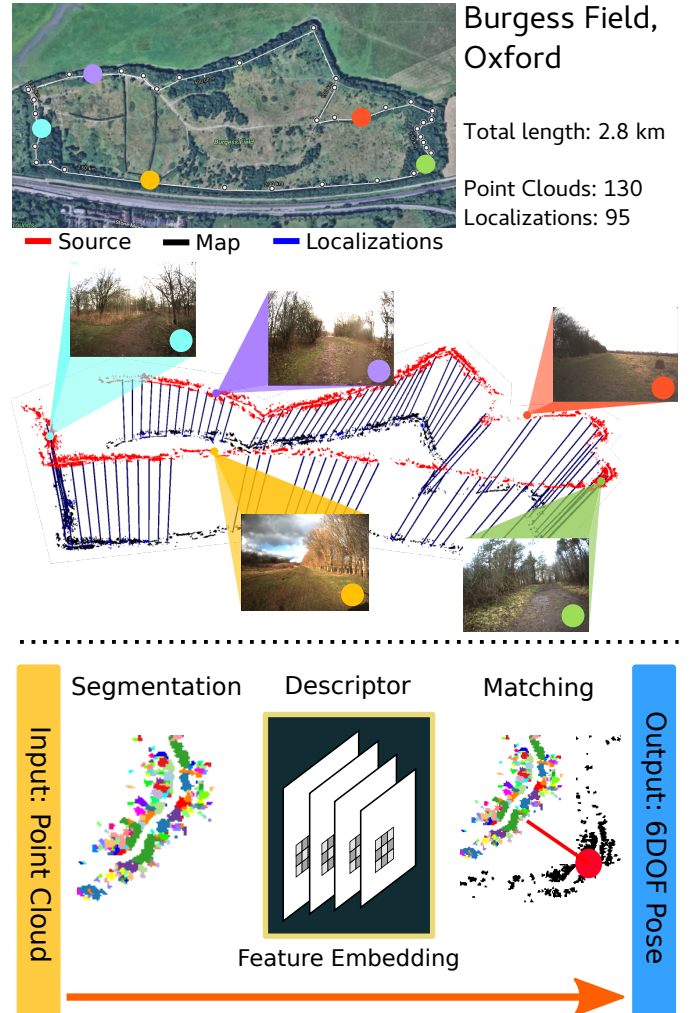


Fig. 1: **Top:** Depiction of the obtained results on large, unstructured environments. The proposed approach regularly proposes correct loop closures in these challenging natural scenes, using a deep learning architecture running only on a CPU. **Bottom:** Overview of the proposed methodology. Segments are extracted from the input point cloud (coloured section) and the map cloud (in black on top). Our neural network features are extracted for all segments. The features from the map are stored in a database. The features from the query cloud are matched against the database. The final pose is then estimated with PROSAC using both the probability of a segment matches (from the network) and the position of the segments (from the map).

planar structure. Our previous work showed promising results in both structured (urban) and unstructured (natural) environments [7]. While reliable localization was achieved, we relied on a hand-crafted set of features which limited the

performance.

The main contribution of this work is a novel description method for segment-based LIDAR localization. We aim to improve upon previous works by learning segment representations in a way that inherently handles the variability of the given environment. The proposed approach learns a descriptor space which efficiently represents the similarities between partial observations of the same segment which makes it robust to incomplete data. We use a neural network to learn this high dimensional feature space. The proposed approach utilizes the convolution operation proposed in [8] to learn an embedding space for both urban and natural scenarios directly from the raw point cloud data. The neural network has the following characteristics:

- Unordered point clouds as input: does not require a specific ordering of the point clouds in a segment. This makes our approach flexible by avoiding the computational cost of creating specific structures for the point cloud and aligning the inputs to a grid or voxelization.
- Feature space is capable of being generalized: experimental validation has proven that our proposed deep learning solution creates a feature space that can generalize, without the need to be retrained to a new sensor or a different environment.
- The network can estimate the quality of a match: a probability is computed and can be used when carrying out probabilistic geometric validation such as PROSAC [9], making our approach more efficient.

In the results section we demonstrate that our proposed method significantly outperforms other hand-engineered approaches, while also improving the computational speed in comparison to other deep learning approaches. In particular we achieve localization performance similar to SegMap [6], but do not require a GPU at run time. Instead the method can be deployed online on the CPU of a mobile robot. This performance improvement comes in large part because our network is more specifically tailored to the task of localization. As presented in [6], the SegMap learning approach can be used to compress and reconstruct point clouds as well as extract and use semantic meaning from the segments to aid localization. We demonstrate performance in urban environments as well as natural environments so as to demonstrate our approach's generality and robustness.

II. RELATED WORK

Robot localization has been heavily explored using different sensors such as LIDAR [10], vision [11], GPS [12] or radar [13] with reliable approaches often combining different sensing modalities [10], [12], [13]. Our proposed work focuses on global LIDAR localization particularly applied to unstructured environments rather than incremental localization or odometry. Here we will briefly overview some methods performing LIDAR localization.

For self-driving vehicles many approaches have taken advantage of LIDAR reflectivity to achieve precise localization. These methods are commonly used in commercial approaches, for example, the approach of [14] uses a prior

map of reflectivity and exploits road marks to reliably localize a vehicle in an urban environment. The authors formulate the world as a mixture of Gaussians (GMM) over 2D grid cells. The GMM represents the heights of points in each cell and the reflectance in a vigorous way that allows the approach to be robust to weather alteration and road degradation.

When localization is carried out concurrently with mapping it is often treated as a research field of its own (Simultaneous Localization and Mapping, SLAM) where the key problem of loop-closure detection is often equivalent to global localization. In this context the robot's odometry provides a rough estimate of the position of the robot within the world which can be leveraged to avoid searching for loop closures over the entire map, thus keeping the computational cost low [15]. When the assumption of a local neighborhood is lost most of these approaches require ICP alignments or similar costly alternatives [16].

Recent approaches aim to localize on a higher level than on explicit point-to-point basis. The concept of segment localization was presented in [17] but more specifically our work is directly motivated by the works of Dubé *et al* [5], [6]. We will describe the framework of segment-based localization in Section III. Features created from LIDAR data are less informative or unique than visual features, thus approaches often prune the candidate matches to reduce the percentage of outliers before performing a robust geometric validation [18]. One of the key advantages of using segments instead of keypoints is that more semantically meaningful entities are extracted, increasing the repeatability of the descriptors. This was demonstrated in [5] and was applied to urban environments. In [6] segment localization evolved into a segment-based SLAM that introduced a new segment feature that also encoded semantics and volumetric shape. This interesting line of work learns features suitable for localization of voxelized segments and the semantics of the map simultaneously. The approach was evaluated in realistic urban setting and in derelict buildings. Elbaz *et al* [19] used segments produced from a Random Sphere Cover Set - overlapping point cloud spheres where each point of the original point cloud could be part of more than one sphere. These spheres were then projected to 2D depth images and processed by a deep auto-encoder. In our previous work [7] we proposed a larger hybrid feature descriptor which improved detection performance and achieved localizations in unstructured environments such as forests, while also being capable of handling the urban scenarios.

The main drawback of classical approaches that use LIDAR data is that they rely on handcrafted features which do not perform as well as learned features [20]. When trying to apply learning techniques to 3D data there are two prominent approaches - representing the data as a voxelized grid, or by using the more common representation of point clouds. Initially these approaches tended to use voxelized inputs as the tensor operators generalize directly to this representation and achieve promising results as shown in [21]. The main drawback of this approach is the amount of computation

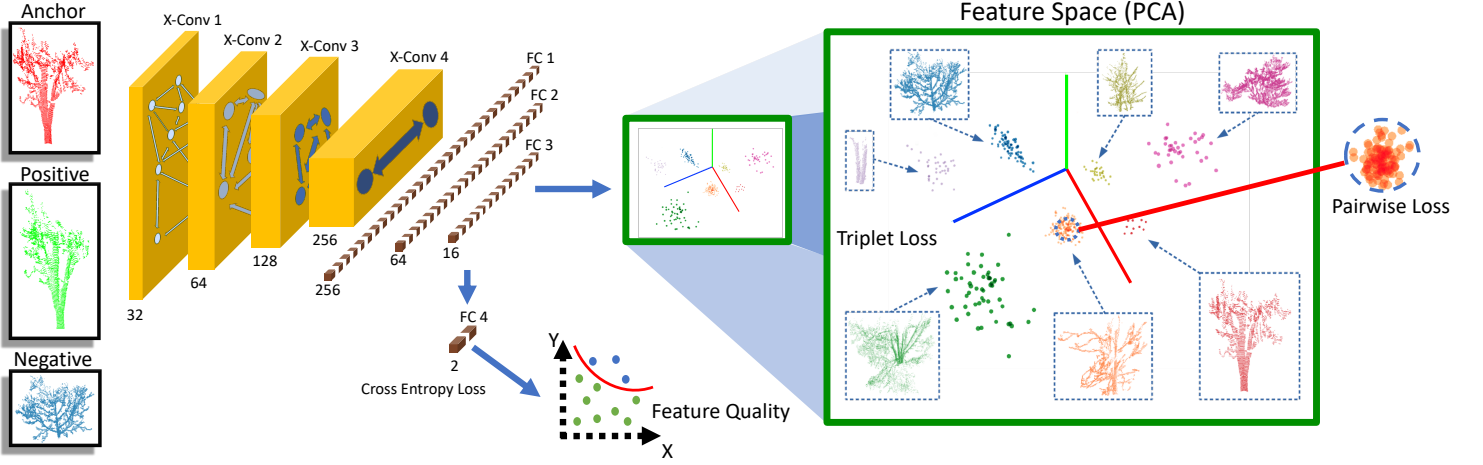


Fig. 2: Network architecture used for learning the feature embedding (left) and a visualization of the embedding space after training (right). The network takes as input raw point data, it uses four x-convolution layers ($\mathcal{X}$ -conv) and three fully connected (FC) layers to estimate the feature space. A feature quality branch is also created by appending a fully connected layer to the feature embedding. The networks learns to cluster similar objects (denoted in the same color) close together while separating dissimilar objects. The final feature descriptor is trained using a combination of triplet and pairwise losses.

required to model the entirety of the environment. In [21] the authors used a $32 \times 32 \times 32$ voxelized grid, resulting in a dimensionality of 32768, with 3D convolutions applied in strides. Because the input resolution is low (due to the computational complexity), these approaches are not capable of handling detailed shapes such as the legs of a chair, for example, as shown in [22]. The issue of handling point clouds was addressed in [23], [24] by learning a symmetry function approximated by a multi-layer perceptron (MLP). The authors created a neural network layer that was invariant to point order. In contrast to the aforementioned volumetric approach, a raw processed point cloud containing 1024 points resulted in a dimensionality of only 3072. In addition, by using a MLP as the basis of their layer and by aggregation of spatial data, Li *et al* [8] managed to create a more descriptive layer to handle unordered point clouds.

In [25] a query point cloud is down-sampled and matched against a collection of previously collected sub-maps globally. All clouds are described on a GPU using a combination of networks trained with a lazy quadruplet loss to produce a 256-dimensional feature vector. In contrast, we firstly segment the query and map point clouds to produce a small set of segments, local keypoints. The segments are then described in real time on a CPU. Thus, our approach uses a combination of local features for each segment and matches those in the map. In this way we are robust to disturbances from occlusions and changing environments.

III. METHODOLOGY

In this section we present Efficient Segment Matching (ESM). This work retains the approach to pose estimation which was initially proposed by [5]. Our contributions in this paper are 1) learning a novel segment descriptor, 2) together with a learned per descriptor measure of performance which enables the pruning of the features before matching and 3) the use of a probabilistic robust pose estimation [9] to improve matching performance.

The problem is formulated as follows: given an input point cloud $\mathcal{P} = \{p_i \in \mathbf{R}^3\}$ and a prior map $\mathcal{M} = \{p_j \in \mathbf{R}^3\}$, we wish to estimate the pose of $\mathcal{P}$ in the map $\mathcal{M}$. We split the task into four modules - Segmentation, Description, Matching and Pose Estimation. Our methodology is illustrated in Fig. 1, bottom. An input point cloud is first segmented using Euclidean segmentation to produce a number of 3D point cloud objects based on the distance of points between them.

Next the segments are passed to a neural network to extract a descriptor of each segment and a measure of quality of that descriptor. For each segment in the live point cloud the N -closest neighboring segments in the map are found based on the Euclidean distance of the descriptors. These matches are ordered based on the extracted feature quality and passed to PROSAC [9] to produce a set of possible localizations. In this work the ordering is done by computing the joint probability of both features (the feature from the live point cloud and the one from the map). The order of the matches is important because the geometric consistency and PROSAC consider first the matches with higher likelihood. Each module of this pipeline is described in depth in previous works [7], [5]. In the following section we detail the architecture and the learning approach.

A. Architecture

Our network architecture is constructed based on the novel convolutional layers presented in [8]. Fig. 2 illustrates our model. The network learns directly from point cloud data, it takes as input a batch of raw point cloud segments. Each segment is uniformly down-sampled to 256 points, zero-centered with normalized variance.

The network used for our descriptor learning approach consists of four $\mathcal{X}$ -conv operators [8] and three fully connected layers; dropout [26] of 0.5 is applied at the second fully connected layer. The outputs of the last fully connected layer are used as the descriptor. The $\mathcal{X}$ -conv operator con-

volves local regions, similar to convolutions in images by a CNN. For each point, its closest N neighbors are projected to a local coordinate frame and lifted to a higher dimensional space with a multi-layer perceptron (MLP). The $\mathcal{X}$ -conv operator learns a convolution based on the MLPs of the neighboring points. In order for the top $\mathcal{X}$ -conv operators to see a larger portion of the point cloud a dilation rate (D) is applied [27]. In this way the receptive field of the top layers is increased without an increase of the neighboring point size (N) or the kernel size.

We have selected a specific configuration of the dilation and neighborhood in each of our layers that better represents the problem we are trying to solve. The first layer uses a neighborhood of 8 and a dilation of 1 per $\mathcal{X}$ -conv operator, the second a neighborhood of 12 and dilation 3, the third a neighborhood of 16 and a dilation of 3 and the fourth a neighborhood of 16 and a dilation of 4. The reason for this is that the first layer will look at a few points (neighborhood 8) that are immediately close (dilation 1). The second will look into more points (neighborhood 12) further out (dilation 3). This way the network creates a representation that aggregates the information slowly creating a hierarchical representation of the whole segment. This is done for each point so as to create a feature of the entire segment from the relative viewpoint of each point. The final descriptor is computed as an average of the features for all the points in a single point cloud. In this way, the feature fuses data from different points creating an expressive yet simple representation. In our experiments we managed to achieve good performance by using an embedding size of 16 dimensions. The dimensionality of a descriptor has a strong impact on the performance of matching. This keeps the computational cost low even if the segmentation algorithm produces many more segments.

The proposed network architecture also includes a classification branch that estimates a measure of the quality of each descriptor. To be able to train this branch we need to train the feature network first. When training the classification branch the descriptor layers are not modified, the model optimizes a single fully connected layer of size two. This represents a logistic regression that classifies whether a feature is good for matching or not. The quality of the feature is determined during training if a successful match is found within the first K neighbors in the dictionary. The classification branch is trained until its accuracy converges at $\approx 70\%$ with $K = 1$ neighbor. At test time the confidence score from the classification branch is used to compute a joint probability of all matched segments. Since two matched segments are gathered during independent observations, we simply multiply the probabilities. During the last stage of the localization pipeline the candidates are sorted using the joint probability distribution and the pose is estimated using PROSAC [9] with applied geometric consistency constraints [7].

B. Learning the Segment Feature Space

To learn the feature space in the proposed architecture a variation of the triplet loss [28] is used. The triplet loss clusters together samples in the feature space that have been

labeled as similar and tries to separate samples that have been labeled as different. By applying the same label to different examples of a segment we introduce invariance to many factors such as noise in the measurements or incomplete segments. The triplet loss defines a triplet as a combination of a sample (anchor) with other two samples, one with the same label and the other with a different label (Fig. 2, left). A pairwise term is defined as a combination of a sample (anchor) with a different sample that has the same label. To train our descriptor we use a variation of the triplet loss similar to the one defined in [29] - in our case we use a squared L_2 norm as shown in Eq. (1) while they use a regular L_2 norm. To train the model we use a batch size of 256 point clouds from which we extract a large set of triplet and pairwise terms.

$$\mathcal{L} = \mathcal{L}_{\text{triplets}} + \alpha \mathcal{L}_{\text{pairs}} + \lambda \|w\|_2^2. \quad (1)$$

During training we identify ≈ 300 times more triplet than pairwise terms. We balance this effect using the α parameter. In Eq. (1) w denotes the parameters of our model, weighted by $\lambda = 10^{-6}$. The proposed architecture contains only 300 K trainable parameters. We use an initial learning rate of $\eta = 10^{-3}$, decaying to minimum $\eta = 10^{-6}$ with ADAM [30] as optimizer. We describe the two losses in detail below.

The classification network is trained using the same data used for the embedding layer. The features are pre-computed and used as a dictionary. To train the classification branch, for each sample we find the K closest matches in the dictionary. A binary label is assigned for each sample: if a match is found in the dictionary it is labeled as 1, otherwise 0. The network optimizes a softmax cross-entropy loss for the classification branch, given the aforementioned labels.

C. Triplet Loss

As commented before, we use a variation of the definition of triplet loss similar to the one in [29]. This loss modified the original triplet loss to solve the vanishing gradient problem. We modify this loss by using squared L_2 norm, instead of the standard L_2 norm, as it created a better clustering of the feature space. The final triplet loss is defined as the sum of the following cost function over all the triplets:

$$\mathcal{L}_{\text{triplets}} = \sum_{(s_i, s_j, s_k) \in \mathcal{T}} c(s_i, s_j, s_k) \quad (2)$$

$$c(s_i, s_j, s_k) = \max \left(0, 1 - \frac{\|f_w(x_i) - f_w(x_k)\|_2^2}{\|f_w(x_i) - f_w(x_j)\|_2^2 + m} \right) \quad (3)$$

$\mathcal{T}$ denotes the set of all possible triplets. s_i and s_j are segments with the same label, while s_i and s_k are dissimilar. $f_w(x_i)$ is the output of the last descriptor layer for an input point cloud x_i and m is a margin regularizer. The latter denotes the minimum ratio for the Euclidean distances between dissimilar pairs of point clouds and similar ones. We use $m = 0.01$ in our experiments. In this manner the triplet loss will cluster similar objects together, while separating dissimilar ones to be farther apart in the feature space. The proposed architecture will provide a descriptor of the whole

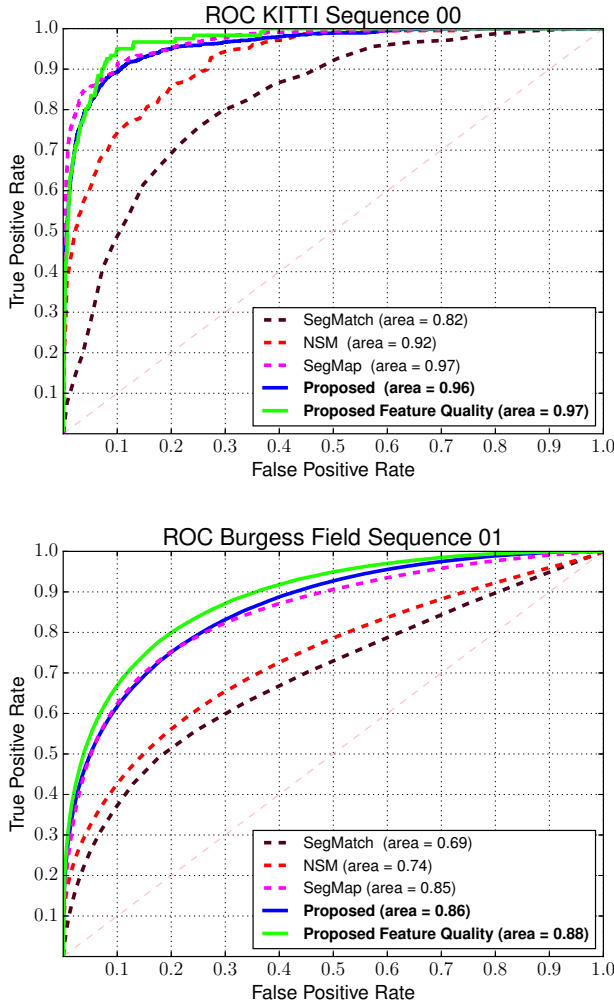


Fig. 3: Receiver Operating Characteristic (ROC) curves for the proposed and baseline approaches [5], [6], [7] on urban (top) and natural (bottom) datasets. The proposed approach performs comparably to the current state of the art on both environments. Learning approaches outperform hand-engineered ones in both urban and natural scenarios.

segment per point, centered at each of those points. Our feature is computed as the average of those descriptors. Our losses are computed using the averaged features. Fig. 2 (right) presents a PCA visualization of the feature space after the network has been trained with the corresponding segments. Different instances of the same tree are clustered together while other trees form their own clusters.

D. Pairwise Loss

The pairwise loss minimizes the distance between samples of the same class (Fig. 2, orange). As proposed by [29], we optimize the following:

$$\mathcal{L}_{pairs} = \sum_{(s_i, s_j) \in \mathcal{P}} \|f_w(x_i) - f_w(x_j)\|_2^2 \quad (4)$$

where $\mathcal{P}$ denotes the set of all pairs. The loss aids the training process by generating very tight clusters, which in turn improve the KNN retrieval during the matching phase.

IV. EXPERIMENTAL EVALUATION

The main focus of this work is to utilize learned features in point clouds to localize with respect to a prior map. The experiments are designed to support our key claims:

- A novel learned feature descriptor that generalizes across a variety of natural and urban datasets.
- Real time operation on a CPU as a result of a compact network architecture.
- The approach produces a measure of quality for each feature which can be used during the pose estimation step to decrease computation.

We provide comparisons against a popular method for segment-based localization, SegMatch¹, as proposed in [5], a data-driven incremental approach SegMap¹ [6], and our previous work NSM [7]. As a metric to evaluate the approaches we have chosen to compare the True Positive Rate (TPR) against the False Positive Rate (FPR) for each classifier, the number of localizations on each dataset, and the computation time for each pipeline. Supplementary material about our model’s hyper-parameters and a video accompany the paper at <http://ori.ox.ac.uk/esm-localization>.

A. Datasets

We perform evaluations using three datasets of our own which were collected in natural environments as well as on a publicly available dataset. First, we compared our novel descriptor against the baseline approaches using the KITTI [4] dataset (Karlsruhe, Germany), taken by a 3D Velodyne-HDL64 sensor in an urban scenario. We use Sequence 06 and 05 to train all descriptors, and Sequence 00 to test them. We also compare the localization performance on Sequence 00. Second, we compared localization performance in a park and forest datasets - George’s Square (Edinburgh, Scotland) and Cornbury Park (Oxfordshire, UK). The former was captured by a Clearpath Husky UGV equipped with a SICK LMS511 LIDAR, while the latter was captured by a vehicle equipped with a Velodyne HDL32E. The datasets are described in detail in [7].

Finally, we used a dataset collected in a natural environment, located at Burgess Field (Oxfordshire, UK). The dataset was captured by a Clearpath Husky UGV equipped with a Bumblebee3 forward facing camera, a Velodyne VLP-16 LIDAR and a Push-broom LIDAR LMS151. The vehicle traversed the same loop of ≈ 2.8 km twice a month during a span of six months. We used the Velodyne data of Sequence 02 and Sequence 01 for training and testing the classifiers. These two sequences were collected two weeks apart each other in February when the foliage was shed. We built a prior map from the Push-broom LIDAR of Sequence 02 using [2]. The source swathes were built using the VO method [2] for every 22 meters the vehicle traversed in Sequence 01.

To label the data we combined four consecutive Velodyne scans using the motion estimator from [2] and extracted segments using distance-based segmentation algorithm. All segments to be within 0.5 m in consecutive point clouds are

¹We use the open-source implementations of SegMatch/SegMap.

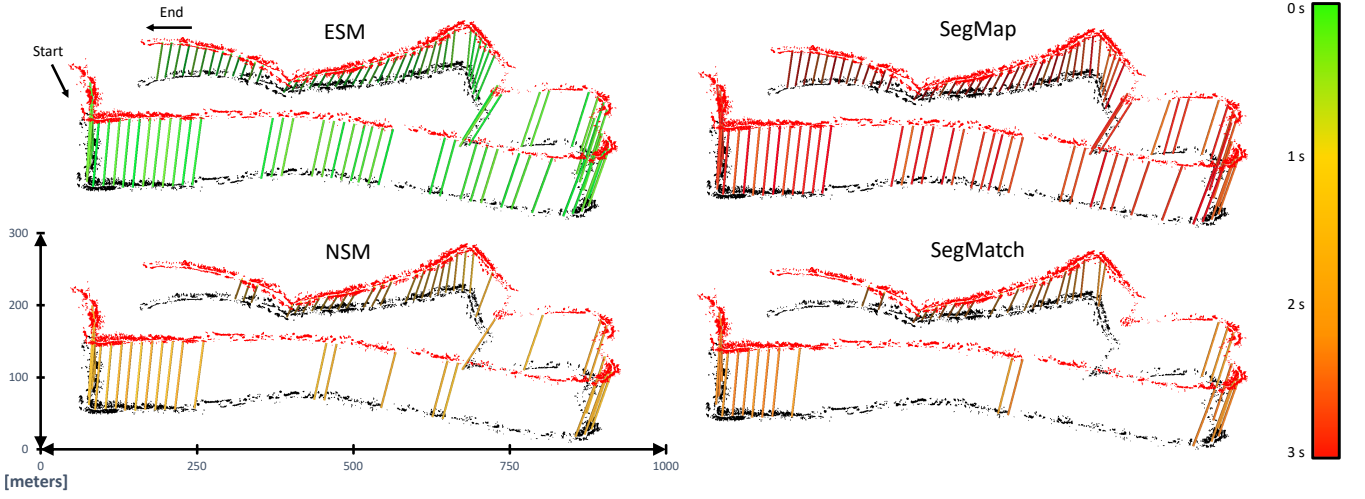


Fig. 4: Illustration of the localization performance of our approach (top left) compared to the baselines on the Burgess Field dataset. The current point clouds (red) are registered to a previous map (black). Each vertical line corresponds to a recognized place, while its color corresponds to the time taken to perform the localization.

Dataset Characteristics					Number of Localizations			
Name	Type	Length (meters)	Sensor	Num. Clouds	SegMatch	NSM	SegMap	ESM
KITTI	Urban	1300	Accum. Velodyne-HDL64E	134	41	53	62	63
George Square	City Park	500	Push-broom	30	8	9	8	11
Cornbury Park	Forest	700	Velodyne-HDL32E	1125	N/A	29	231	248
Burgess Field	Forest	2800	Push-broom	130	41	54	95	94

TABLE I: Number of localizations across one urban and three natural datasets. The proposed approach performs comparably to the state of the art. These experiments also show the ability of the learning approaches to handle data from different types of LIDAR sensors.

considered the same object and labeled as such. All segments further than 4.0 m apart are considered a non-match. In this manner we extracted a total of 82871 segments across 38714 classes for training and 45409 segments in 20583 classes for testing. Training our model on Burgess Field takes 3 hours and 25 minutes from scratch. We have used this model to evaluate our approach on all natural datasets. For experiments carried out on urban datasets, we trained a different model on KITTI in 11 hours and 45 minutes. For all the experiments and methods we use Euclidean segmentation. We segmented the data into individual point clouds depending on the sampling density: 3D Velodyne data: 200–15000 points due to higher frequency, Push-broom type LIDAR: 200–50000 points due to higher density.

B. Classifier Performance

The first experiment evaluates the performance of our model, with and without the feature quality, on KITTI and Burgess Field datasets. Fig. 3 illustrates the Receiver Operator Characteristic (ROC) curves for each of the algorithms. The ROC curve for the feature quality network is created by removing the top *bad* matches prior to evaluation. A performance decrease is seen between the two datasets for all algorithms. We attribute this to the more challenging natural structure in the second environment. In brief, our learning approach generalizes better than engineered features (NSM, SegMatch) across datasets and performs comparably to the learned approach (SegMap). Pruning the matches based on the feature quality shows an improvement with respect to the basic features. For all models we have set the

classifier thresholds at FPR=0.1 for urban and FPR=0.2 for forest environments. We did not retrain the models for each forested scenario or different sensor modality. This shows the proposed features can generalize between datasets and sensors.

C. Localization Performance

In the next set of experiments we aim to support our claim that the proposed algorithm performs comparably to the state of the art on both urban and natural environments when localizing, while requiring less computation. Tab. I shows the total number of loop closures detected in each of the datasets. For each algorithm we have optimized the parameters to retrieve the highest number of true localizations, while having zero false localizations. The learning approaches have not been retrained for each individual natural dataset but still manage to detect about two times more loop closures than the engineered methods. In brief, our algorithm has similar accuracy to [6] on all datasets. Fig. 4 presents qualitative results for our approach compared to the baselines on the Burgess Field dataset, while also highlighting the computational times for each approach. In this environment the vehicle regularly detected loop closures during more than 2 km of operation. Fig. 5 demonstrates the performance of our algorithm across all the datasets. The method did not produce any false localizations, while operating in real time on a CPU.

D. Computation Efficiency

Finally, we evaluate the computational performance of the various approaches. We tested our approach on a mobile Intel

CPU Multi-core execution times								
Algorithm	Desc. size	Segmentation	Pre-processing	Descriptor	Matching (K)	Pruning (RF)	Pose Est.	Total
SegMatch	7 (647)	17	0	605	50 (200)	755	64	1491 ms
NSM	66	17	0	24	207 (200)	913	65	1226 ms
SegMap	64	17	15	5902	19 (25)	0	21	5974 ms
ESM	16	17	3	578	8 (25)	0	9	615 ms
CPU Single-core execution times								
Algorithm	Desc. size	Segmentation	Pre-processing	Descriptor	Matching (K)	Pruning (RF)	Pose Est.	Total
SegMatch	7 (647)	21	0	596	70 (200)	773	64	1524 ms
NSM	66	21	0	37	213 (200)	936	65	1272 ms
SegMap	64	21	26	25945	26 (25)	0	21	26039 ms
ESM	16	21	3	2126	12 (25)	0	11	2173 ms
GPU Computation comparison								
Algorithm	Desc. size	Segmentation	Pre-processing	Descriptor	Matching	Pruning (RF)	Pose Est.	Total
SegMap	64	17	15	15	19	0	15	81 ms
ESM	16	17	3	2	8	0	6	36 ms

TABLE II: Average computational times in milliseconds recorded per point cloud on the Burgess Field dataset.

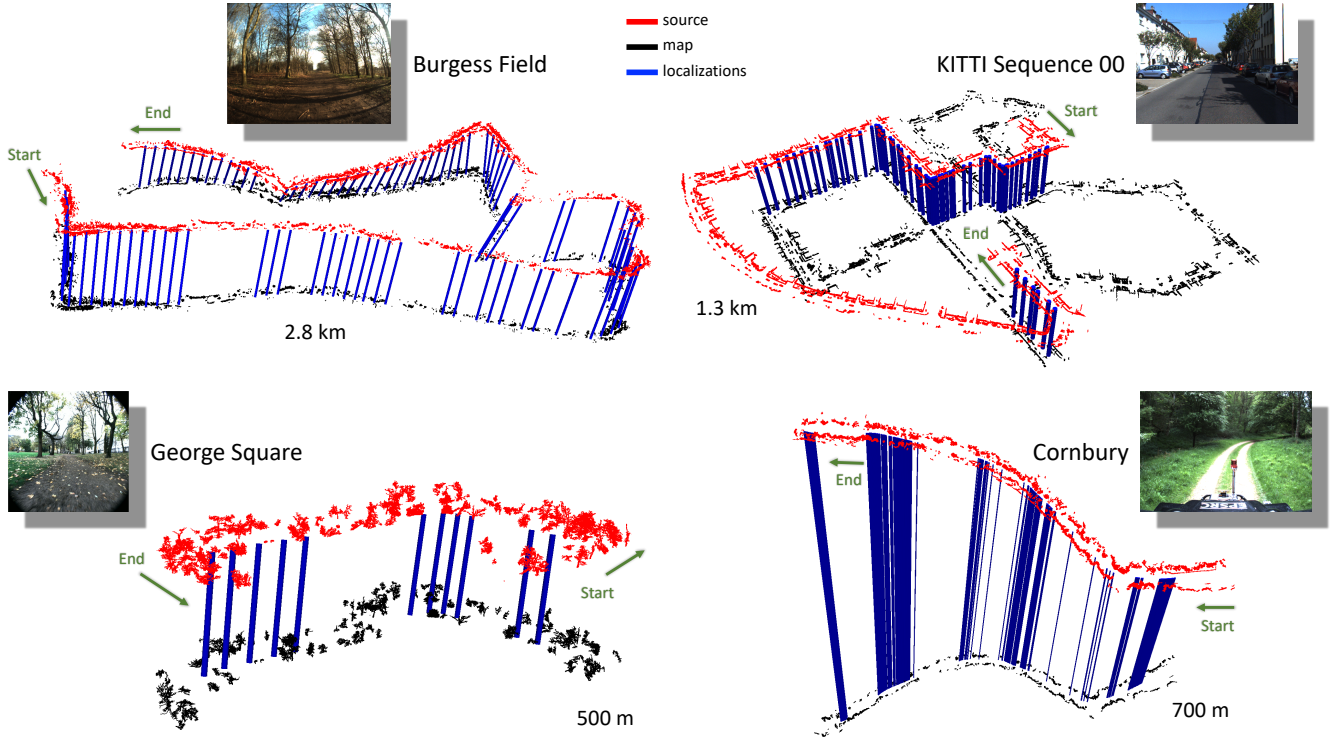


Fig. 5: Qualitative visualization of the performance of the proposed approach (ESM) on all datasets. The used datasets provide a variety of contexts to show the behavior of ESM when coping with structured and unstructured locations.

Xeon E3-1535M CPU with the following configurations: (1) single-core of the processor, (2) multi-core on the same processor, (3) CPU + NVIDIA Titan Xp GPU. We have processed the push-broom point cloud data from Burgess Field and recorded the mean computational speeds. The target map consisted of 2783 segments — individual trees, bushes, and interleaved shrubs. Individual source clouds were created for every 22m traveled with each source cloud containing an average of 46 segments. In order to provide a fair comparison, we have utilized the same Euclidean segmentation method across algorithms, to create the same segments. This is important for the description stage of each pipeline. The segmentation was carefully parameterized to increase localization performance in this challenging natural environment. To estimate the pose we have used the same geometric consistency algorithm for all baselines substituting

RANSAC [31] with PROSAC [9] for the proposed method to incorporate the quality of the features. In this setting, using an incremental geometric consistency as in [6] is not possible, as the push-broom LIDAR provides only a single observation of a segment.

We have empirically evaluated the number of neighbors in feature space (K) on Burgess Field to retrieve the most positive localizations, while keeping zero false positives. For this experiment K influences the KNN retrieval and pruning speeds. $K = 25$ worked well for learning approaches as the features were very descriptive, while we kept $K = 200$ for NSM and SegMatch. Tab. II summarizes the average run time performance, per point cloud, in milliseconds. The size of the descriptor dimension corresponds directly to the computation time taken to describe a segment. The pre-processing and descriptor times scale linearly with the

number of segments in a live point cloud (≈ 46 in Burgess Field). The time required for matching also depends on the embedding dimension. Point clouds tend to be of similar size, thus the computation time for the segmentation does not vary. The pose estimation depends on the number of segments and their closest neighbors (K) and the pose estimator approach. The total size of the SegMatch descriptor is 647 dimensions, only 7 of which are used during the matching stage. These 647 are compressed efficiently to the 45 processed by the Random Forest. This results in fast KNN retrieval and slower RF pruning. The total size of the NSM descriptor is 66, which are extended to 330 for the pruning stage, making it less efficient for KNN retrieval and pruning. The SegMap descriptor requires PCA alignment and voxelization of the segments as pre-processing steps, after which the forward pass of the model is executed in C++. Even though the implementation is efficient, the 9.3M parameters of the network describe a single segment in 0.5 s on a CPU. Tab. II also summarizes the recorded GPU times for the learned approaches with $K = 25$ during matching.

We focused our analysis on CPU-only solutions due to the particular efficiency of our approach. Our network does not require expensive pre-processing of the segments, the model consists of 4 X -conv layers with an input of 256 points. This results in only 300 K parameters, which represents the number of operations needed to be performed. Compared to other learning methods our network has 30–65x less parameters, which allows the model to work in real time on a CPU. In addition, the embedding dimension is kept to just 16 dimensions which speeds up matching.

V. CONCLUSION

In this paper, we presented a novel descriptor for place recognition based on LIDAR segment matching in both urban and natural environments. Our method exploits an efficient deep learning architecture that operates directly on point cloud data without the need of extensive pre-processing. This allows us to successfully detect loop closures at ≈ 1 Hz while using only a CPU. We implemented and evaluated our approach on four different datasets containing natural forest and parkland as well as urban scenes and provided comparisons to other existing techniques. Our approach operates in real time on a CPU and achieves performance comparable to the state of the art, SegMap [6], which requires a GPU to run in real time. The experiments suggest that our approach can be applied to mobile robots with limited computational power. In future work we are interested in deploying the approach on an ANYMAL quadruped and UAVs both of which lack a GPU.

VI. ACKNOWLEDGMENTS

We would like to thank Oliver Bartlett and our colleagues at the Oxford Robotics Institute for the datasets.

REFERENCES

[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: a versatile and accurate monocular SLAM system,” *TRO*, vol. 31, no. 5, pp. 1147–1163, 2015.

[2] W. Churchill and P. Newman, “Experience-based Navigation for Long-term Localisation,” *IJRR*, 2013.

[3] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, “Building Rome in a Day,” *CACM*, vol. 54, no. 10, pp. 105–112, 2011.

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *IJRR*, vol. 32, no. 11, pp. 1231–1237, 2013.

[5] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. C. Lerma, “SegMatch: Segment based place recognition in 3D point clouds,” in *ICRA*, 2017, pp. 5266–5272.

[6] R. Dubé, A. Cramariuc, D. Dugas, J. Nieto, R. Siegwart, and C. Cadena, “SegMap: 3D Segment Mapping using Data-Driven Descriptors,” in *RSS*, 2018.

[7] G. Tinchev, S. Nobili, and M. Fallon, “Seeing the Wood for the Trees: Reliable Localization in Urban and Natural Environments,” in *IROS*, 2018.

[8] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, “PointCNN: Convolution On X-Transformed Points,” in *NIPS*, 2018, pp. 828–838.

[9] O. Chum and J. Matas, “Matching with PROSAC - Progressive Sample Consensus,” in *CVPR*, 2005.

[10] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, “Monte Carlo Localization for Mobile Robots,” in *ICRA*, 1999.

[11] P. T. Furgale and T. D. Barfoot, “Visual Teach and Repeat for Long-Range Rover Autonomy,” *JFR*, vol. 27, no. 5, pp. 534–560, 2010.

[12] J. Levinson, M. Montemerlo, and S. Thrun, “Map-Based Precision Vehicle Localization in Urban Environments,” in *RSS*, 2007.

[13] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-whyte, and M. Csorba, “A solution to the simultaneous localization and map building (SLAM) problem,” *TRO*, vol. 17, pp. 229–241, 2001.

[14] R. W. Wolcott and R. M. Eustice, “Robust LIDAR localization using multiresolution Gaussian mixture maps for autonomous driving,” *IJRR*, vol. 36, pp. 292–319, 2017.

[15] W. Hess, D. Kohler, H. Rapp, and D. Andor, “Real-Time Loop Closure in 2D LIDAR SLAM,” in *ICRA*, 2016.

[16] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, “Comparing ICP variants on real-world data sets,” *AURO*, vol. 34, no. 3, pp. 133–148, Apr. 2013.

[17] B. Douillard, A. Quadros, P. Morton, J. P. Underwood, M. D. Deuge, S. Hugosson, M. Hallström, and T. Bailey, “Scan segments matching for pairwise 3D alignment,” in *ICRA*, 2012.

[18] C. Zach, A. Penate-Sanchez, and M.-T. Pham, “A dynamic programming approach for fast and robust object pose recognition from range images,” in *CVPR*, 2015.

[19] G. Elbaz, T. Avraham, and A. Fischer, “3D Point Cloud Registration for Localization using a Deep Neural Network Auto-Encoder,” in *CVPR*, 2017, pp. 2472–2481.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *NIPS*, 2012.

[21] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction,” in *ECCV*, 2016.

[22] F. Haoqiang, H. Su, and L. J. Guibas, “A Point Set Generation Network for 3D Object Reconstruction from a Single Image,” in *CVPR*, 2017.

[23] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation,” in *CVPR*, 2017.

[24] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space,” in *NIPS*, 2017.

[25] M. A. Uy and G. H. Lee, “PointNetVLAD: Deep Point Cloud Based Retrieval for Large-Scale Place Recognition,” in *CVPR*, 2018.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *JMLR*, vol. 15, pp. 1929–1958, 2014.

[27] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” in *ICLR*, 2016.

[28] K. Q. Weinberger, J. Blitzer, and L. K. Saul, “Distance Metric Learning for Large Margin Nearest Neighbor Classification,” in *NIPS*, 2006.

[29] P. Wohlhart and V. Lepetit, “Learning descriptors for object recognition and 3D pose estimation,” in *CVPR*, 2015, pp. 3109–3118.

[30] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2014.

[31] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *CACM*, vol. 24, no. 6, pp. 381–395, 1981.

Summary

This chapter improved upon the previous localization system proposed in chapter 3 by adopting state-of-the-art learning technique to segment description. It proposed a novel architecture that learned the similarity between segments without the need of any semantic information or precise annotation. The proposed objective function was based on a triplet of anchor, positive, and negative segments, efficiently distinguished in feature space to learn a set of parameters capable of describing a segment during test time. The design of the network enabled the algorithm to run in real time on a GPU-deprived robots. In addition, this chapter proposed a probabilistic method for geometric verification which takes into account the probability of two descriptors being a match, further bridging the gap for a single feature operating in both urban and natural environments.

One of the key empirical observations was the generalization abilities of deep learning features to both urban and natural environments. This observation was in line with the trending deep learning research at the time, however, we believed it was crucial to test this on real-world challenging natural data. The second important observation of this work was the ability of shallow point-based networks to operate in real time without sacrificing performance. This enabled the potential of deploying deep learning algorithms to small sized robots for robust localization.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Learning to See the Wood for the Trees: Deep Laser Localization in Urban and Natural Environments on a CPU
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Tinchev, G., Penate-Sanchez, A., & Fallon, M. (2019, January). Learning to see the wood for the trees: Deep laser localization in urban and natural environments on a CPU. <i>IEEE Robotics and Automation Letters</i> , 4(2), 1327-1334.

Student Confirmation

Student Name:	Georgi Tinchev		
Contribution to the Paper	- Primary author of the work - Developed the network architecture - Designed experiments - Evaluated the network against state-of-the-art related research methodologies - Evaluated the full localization pipeline in real world scenarios - Performed studies on the effects of different objective functions		
		Date	30/12/2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Dr. Maurice Fallon		
Supervisor comments: The contributions listed above are accurate		
	Date	30/12/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

5

Real-time LIDAR localization in natural and urban environments

Motivated by the performance increase shown in chapter 4, in this chapter we dive deeper into understanding the main computational challenges for models used in our work. This chapter proposes a strategy for minimizing the computation of a deep learning model by reducing the amount of data processed in the inner layer of the architecture. Crucially, this is done without major loss in feature matching performance or accuracy while localizing. This chapter also addresses a common issue with point-based networks related to rotation invariance. We produce detailed ablation studies and test the approaches from chapters 3 and 4 as well as the newly proposed approach on a variety of training and testing data. The analysis also consider different sensor modalities. As a final contribution we align 30 sequences from the Oxford RobotCar dataset [Maddern et al. 2017] and offer it to the community. This chapter presents an efficient solution to the global registration problem named Down-sampled Segment Matching (DSM).

This work was submitted to a special issue of Science Robotics on Deep Learning & Robotics, 2020.

Efficient LIDAR-based Global Localization

Georgi Tinchev,^{1*} Adrian Penate-Sanchez,¹ Maurice Fallon¹

¹Oxford Robotics Institute, Department of Engineering, University of Oxford

*To whom correspondence should be addressed; E-mail: gtinchev@robots.ox.ac.uk.

In this work we explore LIDAR-based global localization in both urban and natural environments and develop a method suitable for online robotics applications. Our approach leverages an efficient deep learning architecture capable of learning compact point cloud descriptors directly from 3D data. The method uses a feature space representation of a set of segmented point clouds to match observations of the current scene to a prior map. We show that down-sampling the inner layers of the network can significantly reduce computation time without sacrificing performance. We evaluate the proposed methods on nine scenarios from six datasets including urban, park, forest and industrial environments. Our experiments demonstrate that the method reduces computation by a factor of three and requires 70% less memory with a marginal loss in localization frequency when compared to state-of-the-art approaches. Crucially, the proposed learning method does not require a GPU at run time and can thus be run on robots with limited computation and power payloads such as drones, quadrupeds or UGVs.

Introduction

Robotic systems are being developed in a large variety of environments ranging from underwater survey (1, 2), aerial inspection (3), indoor autonomy (4), urban navigation (5, 6), and exploration of natural environments (7–9). In all of these environments the ability to localize a robot against a prior map of some variety is studied as it is one of the fundamental capabilities in mobile robotics. In this work we demonstrate LIDAR localization which is robust across a broad range of terrestrial environments such as forests, cityscapes and off-road. To achieve state-of-the-art performance we take advantage of a neural network architecture that can learn the most appropriate representation of the environment while being computationally feasible.

When designing a solution to the localization problem, one needs to address the limited resources available onboard an autonomous robot. In contrast to other fields of engineering or computer science, the robot needs to carry its own computation and power resources. For example, previous work studied the need to co-design a mobile robot’s software and hardware to exploit efficiencies (10, 11). While deep learning solutions have gained popularity due to their performance, it has been difficult to leverage them on smaller light-weight platforms because they frequently require GPU hardware that consumes large amounts of power — 200 Watts for a typical desktop GPU. The solutions we aim for should be able to use the power of deep learning approaches while not imposing the need for a GPU and associated power draw. This has clear advantages as it makes the system more applicable to different robotics platforms.

Robotics systems are constantly evolving and adapting to different scenarios. However, not all robots are suitable for all types of scenarios. Cars are of course the norm for autonomous driving applications. For radiation surveying and industrial inspection legged robots are an emerging solution. Drones are commonly used for 3D reconstruction and mapping. When designing an algorithm it is important to take into account the different sensors these platforms

can carry. In this work, we leverage the generalization ability of deep learning to design a solution for different types of LIDAR sensors with varying density, frequency, field of view, and mode of operation.

In this paper we present an approach for large scale real-time LIDAR localization which is trained off-line using a GPU and deep learning, but requires only a CPU at run time to achieve reliable localization. After briefly summarizing our previous works (*12, 13*), we also present a new and more efficient approach with a more compact learned descriptor. The new method is three times faster than previous approaches, consumes 70% less memory at a marginal performance cost, which makes it suitable for CPU-only robots. We present an extensive evaluation of our new approach by comparing to four baseline methods using six different datasets. In total this represents more than 50 km of testing data in a variety of environments - park, forest, urban and industrial scenery. We present an in-depth analysis of the neural network and its localization performance, as well as measuring the computational run time of our method compared to the state-of-the-art. We carry out an ablation study showing that the performance of the approach generalizes across different LIDAR variants and operation environments.

A secondary contribution of this paper is a major improvement to the Oxford RobotCar dataset (*14*) where we precisely aligned and annotated over 2.5 million point cloud segments to enable our learning approach.

Deep learning relies on such large scale datasets to out-perform hand-engineered features on visual classification tasks (*15*). This motivated the creation of standard datasets for training (*16–18*). However, datasets for large scale place recognition and autonomous driving are rare due to the cost associated with collecting them, as well as the inaccuracies in motion estimation which requires extensive post-processing. So as to develop the learning system described below we carefully aligned the Oxford RobotCar dataset (*14*) to centimeter accuracy. We aligned 30 sequences captured throughout the year totaling 300 km of LIDAR data. We make this

substantial effort on improving the dataset publicly available and is one of the largest annotated LIDAR datasets.

Results

The main focus of this work is to localize a 3D *scene* point cloud with respect to a prior map. In this section we present the individual components of our system. We also overview several baseline approaches before comparing performance using six challenging datasets.

Figure 1 illustrates qualitative examples of the performance of the proposed algorithm on six challenging datasets, ranging from urban environments to industrial facilities and heavily vegetated forests. The figure also previews the 30 aligned sequences from the RobotCar dataset in Oxford, UK. Finally, it shows the robotics platforms and sensors used in our experimental evaluation.

We compare our method against two popular methods for segment-based localization - SegMatch* (19) and the data-driven approach SegMap* (20), as well as our previous works - NSM (12) and ESM (13). When evaluating our experiments we define the following abbreviations (ordered by publication date):

- SM-Eig - Baseline model, SegMatch (19), trained and evaluated with Eigenvalue features.
- SM-ESF - Baseline model, SegMatch (19), trained and evaluated with Eigenvalue and Ensemble of Shape Histogram features (ESF).
- NSM-RF - Our previous work, NSM (12), with Gestalt features and a Random Forest.
- SegMap - Baseline model (20) with voxel-based deep learning architecture.

*We use the open-source implementations of SegMatch/SegMap at <https://github.com/ethz-asl/segmap>.

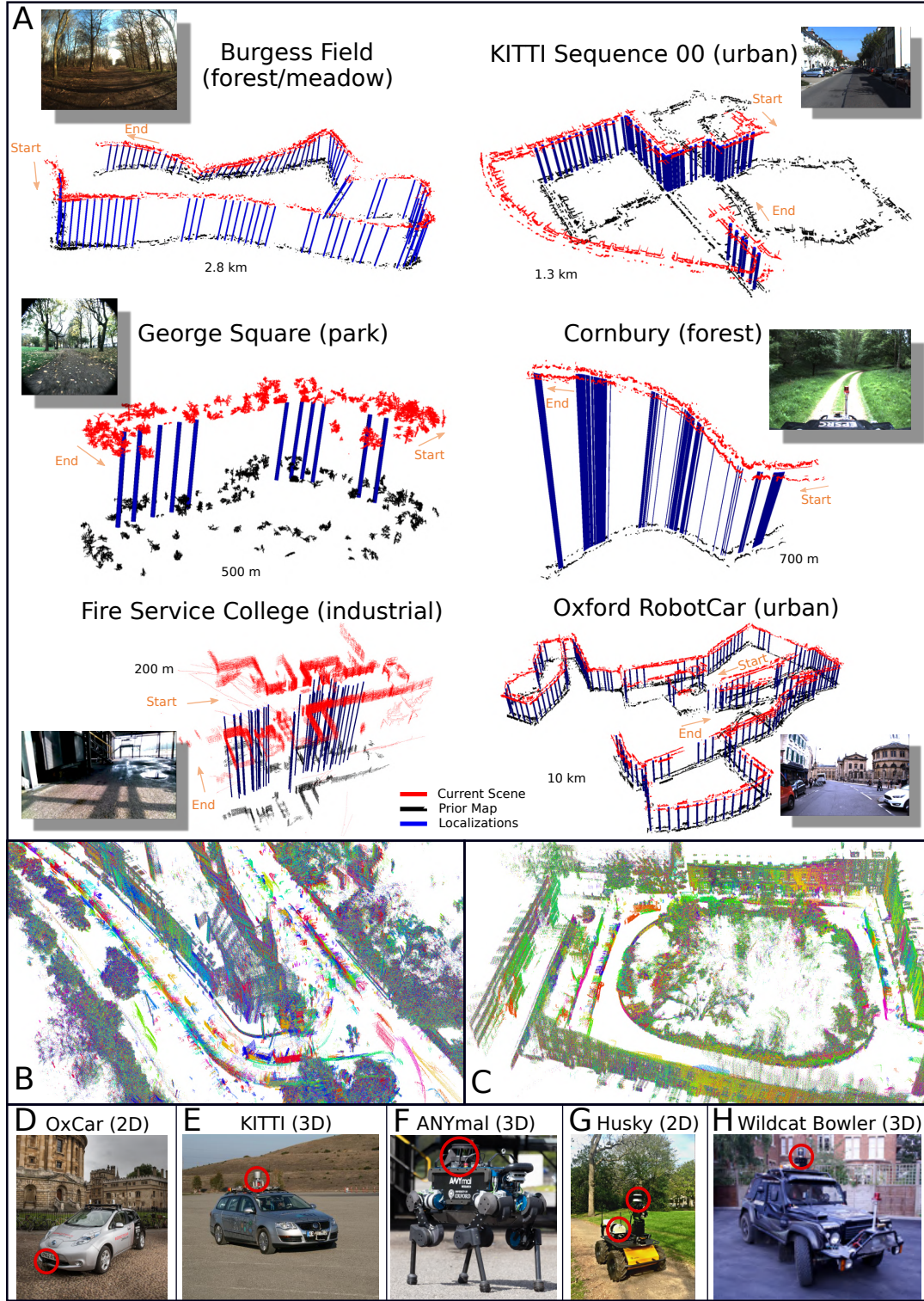


Figure 1. **A.** Top-down visual preview of all the datasets considered in our work. The sequence of *scene* point clouds are denoted in red and have been vertically offset from the map clouds (in black) in Z for visualization purposes. The vertical blue lines indicate a correct place recognition in the map’s reference frame. **B,C.** Preview of the aligned Oxford RobotCar dataset across 30 sequences. **D-H.** Robotics platforms and LIDAR sensors used in our study.

- ESM and ESM-FQ - Baseline, based on our previous method (13) with and without a feature quality (FQ) classification branch.
- **DSM** and **DSM-FQ** - The proposed method with and without the feature quality of the classification branch.

Our approach is modular, because of this we can evaluate each individual component of our system and compare it to the baseline methods. First, we examine the effect of the learned features and their performance on three different datasets - two urban scenarios and one parkland scenario. Second, we present an ablation study that explores the generalization ability of each method by looking at different permutations of training and testing data across the datasets. Third, we address a common issue faced by some approaches relating to rotation variation and viewpoint change. Finally, using the optimal set of system components, we analyze the overall execution of our system both in terms of localization performance and run time efficiency.

Feature Performance

We begin the evaluation by looking at the descriptive power of the learned features. We chose the KITTI (5), Burgess Field (own) and RobotCar (14) datasets to perform feature description. We chose these three datasets due to the large amounts of labeled data and the different point cloud density they offer.

To measure performance we compute the Receiver Operating Characteristic (ROC) curve, which is a standard metric for the true positive rate against the false positive rate for each dataset given a pair of segment descriptors and their corresponding labels.

Figure 2 presents three ROCs for KITTI, Burgess Field and RobotCar datasets. Our previous work ESM (13) typically performs best. The proposed method, **DSM**, is consistently second-best with a difference of 1-2% — but again we emphasize the computational efficiency of

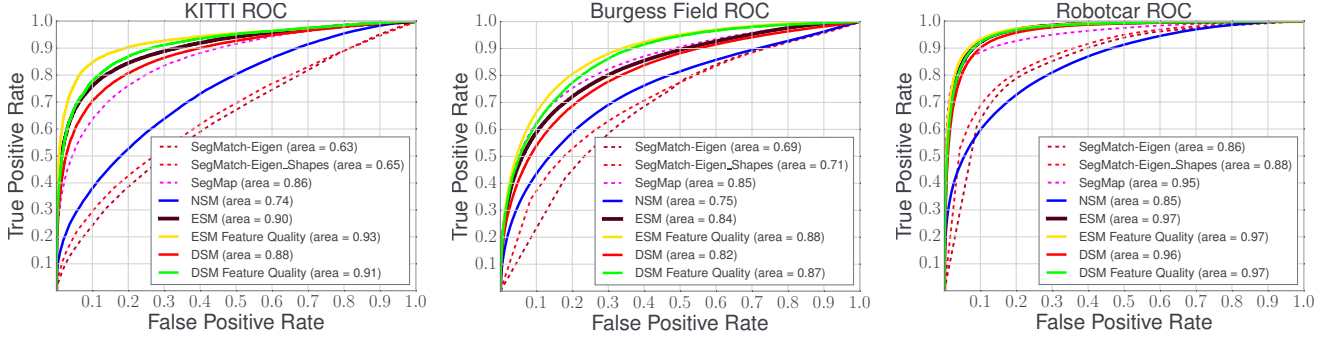


Figure 2. Receiver Operating Characteristic (ROC) curves for the proposed method, **DSM**, and baseline approaches on KITTI (urban, left), Burgess Field (natural, middle) and Oxford RobotCar (urban, right) datasets.

DSM. The classification (Feature Quality) branch is trained specifically to provide additional information about whether two point cloud segments match each other. By directly maximizing this metric, the increase in performance of the feature quality branch is evident. We note that all methods perform slightly worse on the natural dataset (Burgess Field) as compared to KITTI, as the trees and bushes in that scene are more challenging to classify. Similarly, all methods perform slightly better on the RobotCar dataset, due to the better alignment and similarity of segments. In line with the literature, we show that learning methods outperform methods based on engineered features (SegMatch (19) and NSM (12)).

Generalizability

In this section we study the different trained models to test their ability to generalize. We employ the area under the ROC curve (AUC) metric for each combination of training and testing datasets. We trained our baselines and the proposed model to determine how performance degrades when the testing data differs from the training data.

Table 1 presents the AUC for each of the deep learning and Random Forest models. For each testing dataset, the best performing method is indicated with a percentage value of 100% (in green). For ease of reading we compute the loss in performance relative to that best method,

Table 1. Ablation results compared to baselines. The table shows the difference in performance when training with a dataset and testing on another (possibly different) dataset. The best performing algorithm is colored green and the second best orange. Each method reports Area and Percentage, "Area" is the area below the ROC curve that the method obtained when testing on that dataset while "Perc." is the performance relative to the best performing algorithm.

Test	Train	SM-Eig		SM-ESF		NSM-RF		SegMap		ESM		ESM-FQ		DSM		DSM-FQ	
		Area	Perc.	Area	Perc.	Area	Perc.	Area	Perc.	Area	Perc.	Area	Perc.	Area	Perc.	Area	Perc.
KITTI	KITTI	0.632	68%	0.655	70%	0.738	79%	0.858	92%	0.902	97%	0.930	100%	0.880	95%	0.911	98%
	Burgess Field	0.666	72%	0.711	76%	0.736	79%	0.764	82%	0.828	89%	0.888	95%	0.864	93%	0.920	99%
	RobotCar	0.680	73%	0.709	76%	0.752	81%	0.822	88%	0.873	94%	0.890	96%	0.807	87%	0.825	89%
Burgess Field	Burgess Field	0.685	78%	0.714	81%	0.753	85%	0.850	96%	0.836	95%	0.883	100%	0.815	92%	0.869	98%
	KITTI	0.626	71%	0.697	79%	0.713	81%	0.781	88%	0.817	93%	0.868	98%	0.797	90%	0.849	96%
	RobotCar	0.710	80%	0.730	83%	0.732	83%	0.808	92%	0.814	92%	0.853	97%	0.794	90%	0.839	95%
RobotCar	RobotCar	0.856	88%	0.880	91%	0.848	87%	0.953	98%	0.969	99%	0.972	100%	0.961	99%	0.967	99%
	KITTI	0.760	78%	0.821	84%	0.815	84%	0.952	98%	0.934	96%	0.926	95%	0.928	95%	0.929	96%
	Burgess Field	0.819	84%	0.853	88%	0.865	89%	0.951	98%	0.934	96%	0.948	98%	0.922	95%	0.938	97%

expressed again as a percentage (always less than 100%). The second-best method is highlighted in orange.

We note that the Feature Quality branch for the proposed method generalizes much better than the approach without the feature quality classification branch. Regardless of the choice of training and testing dataset, the drop in performance of our model is comparable to the other learning approaches. **DSM** gives consistently the second best performance, trailing just behind our previous method ESM (13) — with much reduced computation, as we will later see.

When evaluating the hand-crafted feature methods (NSM, SM-Eig, SM-ESF) on the KITTI dataset, we found that Random Forests trained on Oxford RobotCar actually outperform the Random Forest models trained on KITTI. We attribute this to the larger sample size of the Oxford RobotCar dataset, as well as the better alignment of the training data. For the learning approaches, the performance drop is 10% in the worst cases. Interestingly, **DSM-FQ**, trained on Burgess Field, performs slightly better than when trained on KITTI. We attribute this small variation to the FQ branch of the architecture.

When testing on the Burgess Field dataset, the proposed method (**DSM** and **DSM-FQ**) has just a 2% drop of performance when trained on KITTI, and 3% when trained on RobotCar. This

is a generalization similar to ESM (13), but better than the other baselines.

On the RobotCar dataset, the trained models of **DSM** and **DSM-FQ** consistently perform second best, by about 1% percent — second to ESM (13). We also note that the scores reported by all algorithms when tested on the RobotCar dataset are consistently higher regardless of the training dataset (inline with the ROC experiment).

In summary, the proposed method, **DSM**, gives performance which is almost directly comparable to the best performing model (typically our previous method ESM (13)) with only a 1-2% reduction in performance.

Rotation Variation

In the previous sections we showed that deep learning methods perform well when trained and tested on the same datasets, but also generalize well to unseen data. In this section we investigate a common issue in point-based networks, such as the one used in (13). These types of architectures struggle with viewpoint variation — when the same scene is sensed from different orientations. This is because robustness to this variation is not typically ingrained within the networks.

We train our descriptor, **DSM**, with rotation augmentation which applies random yaw rotations uniformly sampled between 0 and 180 degrees for each sample. We also further pre-process the individual clouds before inputting them to the network. The computational cost associated with this is negligible as we will see later. We chose to compute the following distance as a metric to evaluate how robust our model is to random rotations: $\delta = d(c_n, c_0)/Z$, where d is the distance in feature space between a segment c_n , where $n \in \{0, 10, 20 \dots 360\}$ represents the features of a segment rotated every 10 degrees and c_0 (the unrotated reference). Z is a normalization factor that represents the mean distance between all other samples in the dataset. Thus, the metric is lower for architectures that are more robust to rotations.

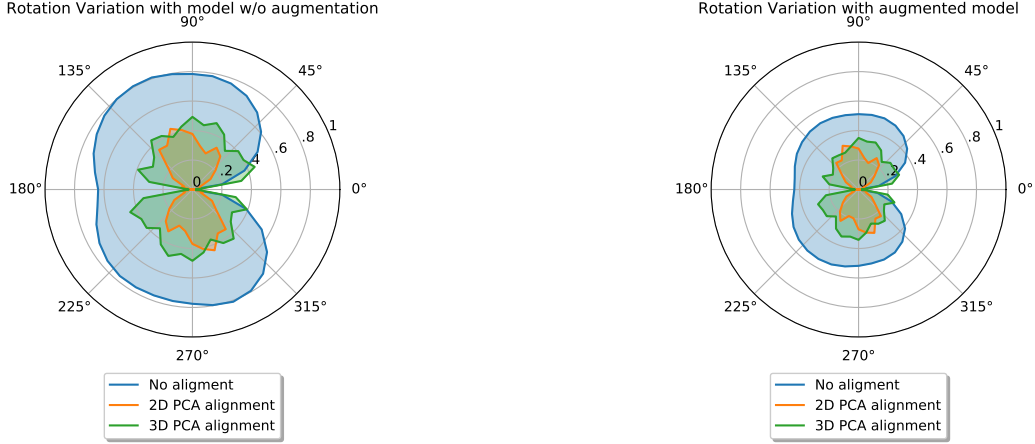


Figure 3. Rotation variation with two models trained without augmentation (left) and with it (right). We have evaluated two alignment strategies based on PCA at different rotation angles. **Lower score indicates better performance.**

Figure 3 presents two different variations of our model - with and without applying rotation augmentation to random segments during training, as well as 3 different modes of pre-processing: no pre-processing (similar to (13)), 2D PCA alignment, and 3D PCA alignment. The angle indicates a sample at different rotations (n), while the radius value corresponds to δ . The result indicates that while augmentation alleviates the issue, pre-processing is essential for mitigating rotation variations. In experiments which follow we utilize 2D PCA alignment pre-processing.

Localization Results

Thus far we have focused primarily on the deep learning part of our approach. In this section we evaluate the complete performance of our method — how well and how often it localizes the current scene within a large prior map.

First, we evaluated an urban environment, consisting of four sequences from the RobotCar dataset and sequence 00 from KITTI. Second, we have considered different types of natural environments: parkland (George Square), forested meadows (Burgess Field), and a heavy forest

Table 2. Number of localizations across all datasets.

Test Set Characteristics					Number of Localizations				
Name	Type	Length	Sensor	Num. Clouds	SegMatch	NSM	SegMap	ESM	DSM
Oxford RobotCar	Urban	50310 m	Push-broom	1118	80	346	915	896	773
KITTI	Urban	1300 m	Velodyne-HDL-64E	134	41	53	62	63	62
Fire Service College	Industrial	190 m	Velodyne-VLP-16E	169	13	12	35	45	46
George Square	City Park	500 m	Push-broom	30	8	9	8	11	10
Cornbury Forest	Forest	700 m	Velodyne-HDL-32E	1125	N/A	29	231	248	208
Burgess Field	Forest	2800 m	Push-broom	130	41	54	95	94	90

(Cornbury Forest). Third, we tested the algorithms in an industrial setting using the ANYmal quadruped at the Fire Service College, Gloucestershire. These datasets not only present different environments, but also different sensor variations. When evaluating the datasets, it should be noted that the total number of clouds does not reflect the number of possible localizations. This is due to the fact the robots sometimes traversed parts of the environments missing from the prior map. Given this, we were pleased that the approaches are robust and did not compute any false localizations in those areas. For the vegetated datasets, and the industrial dataset we used the model trained on the Burgess Field dataset. Therefore, the training and testing data is inherently different as it comes from different sensors. The rest of the datasets used their corresponding training data - RobotCar trained on another 26 sequences, and KITTI trained on Sequences 05 and 06.

Table 2 shows the localization results for all the datasets. Qualitative examples of the performance of our algorithms is illustrated in Figure 1 and Movie S2 in the Supplementary Materials. The proposed method performs commensurately to the other learning approaches — ESM and SegMap.

Computational Efficiency

Having demonstrated typical performance of our proposal approach against both deep learning and engineered approaches, in this section we study the computation cost associated with each method.

We tested the methods on a mobile Intel Xeon E3-1535M CPU with the following configurations: (1) a single-core of the processor, (2) multi-cores of the same processor, and (3) the CPU with a NVIDIA Titan Xp GPU. We have processed the push-broom point cloud data from Burgess Field and recorded the mean computation time as milliseconds per localization query.

To provide a fair comparison, we have used the same Euclidean segmentation method across the algorithms in order to create the same segments, and thus the segmentation time is equivalent. We empirically evaluated the number of neighbors in feature space (K) to retrieve the most positive localizations, while keeping zero false positives - 25 for learned approaches, and 200 for hand-engineered ones. The value of K influences the pose estimation stage as well. We also note that the size of the descriptor affects the time for matching and pruning.

Table 3 summarizes the recorded GPU and CPU times for all the approaches. The proposed method, **DSM**, performs three times faster than the previous best (*l3*) on a CPU, while performing similarly on a GPU. We attribute this to the sampling performed at each layer, as the number of parameters is fairly similar between the two architectures - 280 K vs 300 K. We also note that the cost of pre-processing using PCA alignment is negligible compared to the other components and is justified given the benefits we saw from the Rotation Variation experiment.

Table 3. Average computation times in milliseconds recorded per point cloud for a localization query on the Burgess Field dataset.

CPU Multi-core execution times								
Algorithm	Desc. size [dim]	Segmentation [ms]	Pre-processing [ms]	Descriptor [ms]	Matching [ms] (K)	Pruning [ms] (RF)	Pose Est. [ms]	Total [ms]
SegMatch	7 (647)	17	0	605	50 (200)	755	64	1491
NSM	66	17	0	24	207 (200)	913	65	1226
SegMap	64	17	15	5902	19 (25)	0	21	5974
ESM	16	17	3	578	8 (25)	0	9	615
DSM	16	17	8	184	8 (25)	0	9	226
CPU Single-core execution times								
Algorithm	Desc. size [dim]	Segmentation [ms]	Pre-processing [ms]	Descriptor [ms]	Matching [ms] (K)	Pruning [ms] (RF)	Pose Est. [ms]	Total [ms]
SegMatch	7 (647)	21	0	596	70 (200)	773	64	1524
NSM	66	21	0	37	213 (200)	936	65	1272
SegMap	64	21	26	25945	26 (25)	0	21	26039
ESM	16	21	3	2126	12 (25)	0	11	2173
DSM	16	21	8	731	12 (25)	0	11	783
GPU Computation comparison								
Algorithm	Desc. size [dim]	Segmentation [ms]	Pre-processing [ms]	Descriptor [ms]	Matching [ms] (K)	Pruning [ms] (RF)	Pose Est. [ms]	Total [ms]
SegMap	64	17	15	15	19 (25)	0	15	81
ESM	16	17	3	2	8 (25)	0	6	36
DSM	16	17	8	2	8 (25)	0	6	41

When training **DSM** we noticed that the GPU memory consumption compared to ESM (13) for the same batch size was 70% less. Therefore, to fully utilize the GPU we have increased the batch size when training **DSM**. We also note that the pose estimator computes a transformation estimate quicker, due to the probability information extracted from the classification branch and the use of the PROSAC estimator (21).

Discussion

In this paper we propose a method for global localization that leverages the advantages of deep learning, while also being efficient and ready for deployment on a CPU. Based on our experiments, we have concluded that the proposed approach is able to generalize well in unseen environments and is robust to the choice of LIDAR sensor. By leveraging data augmentation during training and pre-processing during inference, our approach handles unseen rotations as well, at a negligible run time cost. It also performs comparably to the state-of-the-art on the overall localization task. Our method serves as a demonstration that robots with limited computation power, e.g. drones and legged robots, which cannot carry a GPU, can benefit from the advantages of deep learning. This is all possible due to the introduction of efficient point sampling in the inner layers of the architecture, which reduces the number of required parameters and summarizes the features efficiently.

The research in this paper can inspire a number of interesting directions of further work. With the methods presented in this work, researchers could analyze different learning techniques for performing efficient segmentation. This could be developed into portable end-to-end learning methods suitable for mobile operation. This work also provides a framework for a number of other applications such as augmented reality, cost efficient pose estimation, SLAM.

The presented architecture in this work can be extended with other applications in mind. The role of long-term segmentation could be analyzed in a greater context. Currently, our

method handles variations in the shape of each object, but sometimes incomplete segments can cause observation differences, even when inspected by a human. Long-term variations, e.g. a construction site that changes regularly, are currently handled via the geometry verification step of our pipeline. Further research would allow greater flexibility in choosing between multiple versions of the same segment. This *evolution* over short (e.g. a car with LIDAR sensor passing by) or long (e.g. construction site) periods of time presents interesting challenges on its own for the matching process.

Despite extensive research into deep learning for the last 20 years, its operation on a CPU is highly restricted and remains an open research question. Many of today’s advanced robots are not capable of leveraging the benefits of these methods, due to the weight and power requirements of a GPU. While this work investigates efficient deep learning for localization and robot navigation, there are many more areas where efficiency is critical. One such area is the ability to avoid objects during close loop navigation, for which the proposed sampling technique could be applied.

The contribution of our annotated dataset can also aid other data scientists in their research. The alignment of the provided data surpasses any automated method relying on high-end GPS sensors. The data can be used for change detection, season variation, surveying, long-term SLAM, etc. In particular, the provided dataset will enable researchers to develop algorithms with a higher degree of accuracy for outdoor navigation experiments.

Materials and Methods

In this section we present our pipeline for global LIDAR localization. We first present a problem statement formally defining the structure of our approach and then we describe in detail the proposed improvements afforded by our method. Second, we present the different datasets, robotic platforms and types of LIDAR sensors employed in testing. We discuss in detail the

various subtleties of the experimental validation and field trials to motivate and structure the experiments.

Problem Statement

We formulate the global localization problem as follows: given an input (*scene*) point cloud $\mathcal{P} = \{p_i \in R^3\}$ and a prior map point cloud $\mathcal{M} = \{p_j \in R^3\}$, we wish to estimate the pose of $\mathcal{P}$ in the map $\mathcal{M}$, formally defined as ${}^{\mathcal{M}}T_{\mathcal{P}}$. Typically the prior map is quite large (10s of km traveled) while the input point cloud is the local observation around the robot.

We solve the problem by first segmenting the prior map $\mathcal{M}$ into a set of prior map segments $s_t \in \{s_{t1}, s_{t2} \dots s_{tM}\}$, where $s_{tj} \in \{(p_{j0_x}, p_{j0_y}, p_{j0_z}), (p_{j1_x}, p_{j1_y}, p_{j1_z}) \dots (p_{jT_x}, p_{jT_y}, p_{jT_z})\}$. The number of points in each segment, T , are determined by the segmentation method. We then describe each segment s_{tj} with a descriptor function $f(\cdot)$ to produce a set of prior map segment features for each segment, $\mathbf{f}_{tj} \in R^D$, where D denotes the feature dimension. To avoid ambiguity we will use $\mathbf{f}_t \in R^D$ to denote the collection of segment features for a particular point cloud map. Furthermore, we identify a key pose for each prior map segment, $\mathbf{k}_{tj} \in R^6$. The key pose and features are stored in a database for further processing at a later stage.

Secondly, for each *scene* point cloud, $\mathcal{P}$, we segment the cloud into a collection of scene segments, $s_s \in \{s_{s1}, s_{s2} \dots s_{sP}\}$, where $s_{si} \in \{(p_{i0_x}, p_{i0_y}, p_{i0_z}), (p_{i1_x}, p_{i1_y}, p_{i1_z}) \dots (p_{iT_x}, p_{iT_y}, p_{iT_z})\}$. Note that the number of points for each segment in the map, s_t , will not be equal to the number of points in each segment extracted from the scene point clouds, s_s . We use the T for simplicity - in practice the number of points in each segment will vary. We then determine a key pose for each segment in the scene cloud, $\mathbf{k}_{si} \in R^6$, and describe each segment with $f(\cdot)$ to produce a set of scene segment features, $\mathbf{f}_{si} \in R^D$. We will use $\mathbf{f}_s \in R^D$ to denote the collection of segment features for a particular scene point cloud.

Thirdly, a set of candidate matches is created by utilizing a distance function $d(\mathbf{f}_t, \mathbf{f}_s)$, pro-

ducing $((\mathbf{k}_{s_p}, \mathbf{k}_{t_p}), (\mathbf{k}_{s_q}, \mathbf{k}_{t_q})) \in \mathcal{C} = \{(\mathbf{k}_{s_0}, \mathbf{k}_{t_0}) \dots (\mathbf{k}_{s_L}, \mathbf{k}_{t_L})\}$, where $\mathcal{C}$ is the set of accepted matches proposed by $d(\cdot)$ and L is its cardinality.

Finally, ${}^{\mathcal{M}}T_{\mathcal{P}}$ is estimated based on $\mathcal{C}$ by using a robust estimator. This gives the estimated location of the *scene* point cloud within the prior map.

This general method has been used by a number of approaches similar to ours (12, 13, 19, 20).

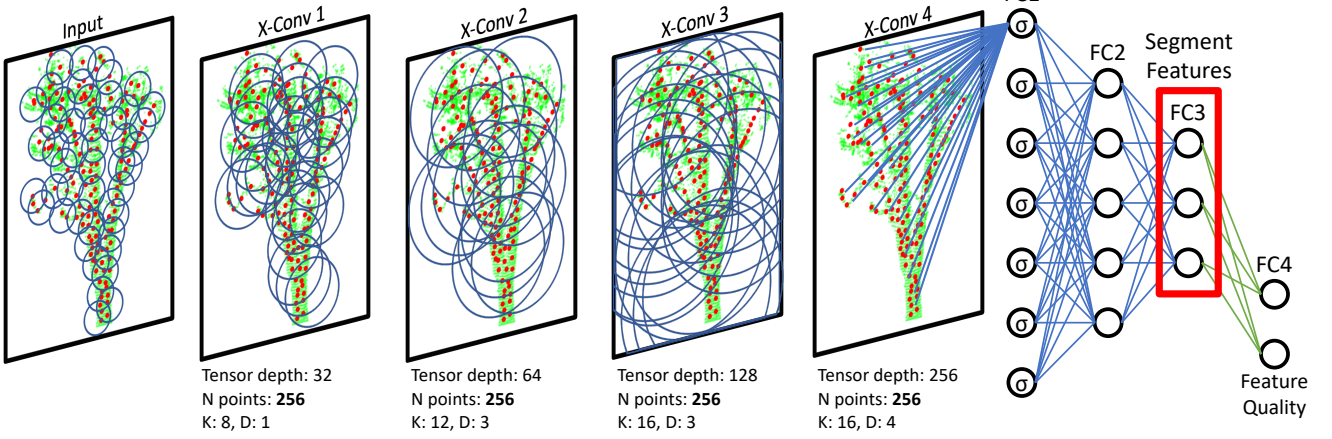
DSM: Down-sampled Segment Matching

In this section we describe our proposed method, named Down-sampled Segment Matching (**DSM**), which has the following properties: 1) a more efficient neural network architecture for point cloud description suitable for both GPU and CPU inference, and 2) a down-sampled segment descriptor at each layer, which achieves 3) three times quicker inference and 70% less memory utilization than the previous best (13). The reason for this is that at each layer of the model presented in (13), the $\mathcal{X}$ -conv operator (22) computes the k-nearest neighbors for each point, which is very expensive. Here, by carefully sampling the points, we reduce the required computation by a factor of three, while retaining the important information for segment matching. In the following sections we describe the architecture and down-sampling in more detail.

Network Architecture

The architecture for **DSM** is presented in Figure 4, bottom. We have opted for a smaller point size per layer, in order to reduce the computation load in comparison to our previous method (13). The input to our network is 256 points. These points are then down-sampled to 160 in the first layer with the number of nearest neighbors $N = 8$. A dilation of 1 is applied, resulting in the layer processing every point from the input point cloud. Intuitively, our first layer reduces the input by 40%. The second layer of the model has a point size of 96 with the number

ESM Architecture



DSM Architecture

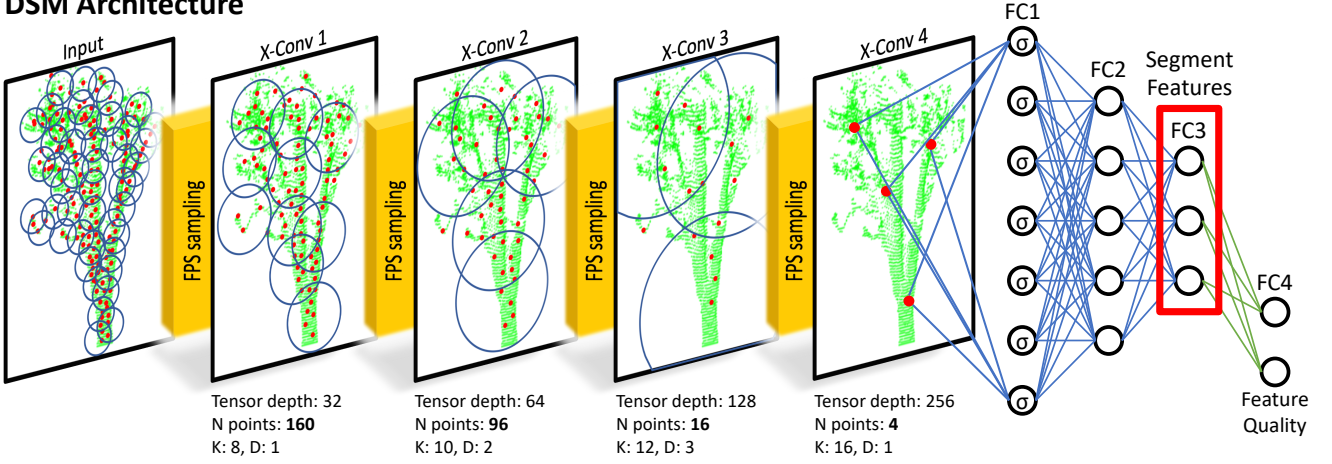


Figure 4. ESM (13) (top) and **DSM** (bottom) architectures. The input size is 256 points. At each layer the number of points (N), nearest neighbors (K) and dilation rate (D) are specified. The descriptor of the network is extracted from the third fully connected layer (FC3). Both architectures have an additional classification branch (FC4), defining the quality of the descriptor. For **DSM**, we utilize efficient Farthest Point Sampling (FPS) at each layer to decrease the number of points seen by the network at later stages.

of nearest neighbors, $N = 10$ and dilation of 2. In this manner we ensure that every point in the new layer is connected to 10 neighbors from the layer above, which are equally spaced, every 2 points. Therefore, a single point aggregates features in a hierarchical way. Similarly, our third layer has a point size of 16, each point is connected to $N = 12$ neighbors, dilated every 3 points. This representation shrinks the number of points by 80% with each remaining point being much more informative. The final $\mathcal{X}$ -conv layer has size 4, with $N = 16$, and dilation of

1. This layer ensures that every point in the last layer *sees* all the possible information, while retaining a small number of points. We have empirically evaluated that this architecture consumes 70% less memory for the same input batch, when compared to our previous work (13), and has slightly fewer parameters (280 K compared to 300 K). We also proposed a different selection methods for points in each subsequent layer including random sampling, inverse density sampling and Farthest Point Sampling (FPS). Our experiments showed that FPS improves the overall results, as it selects points based on coverage. Using this evidence we established solid grounds on which to propose our novel neural network architecture (**DSM**) that reduces the computational cost greatly compared to most computationally efficient baseline (ESM).

Effects of sampling in compact neural architectures

This section focuses on Farthest Point Sampling (FPS) and describes an efficient implementation of FPS suitable for CPU execution during inference.

The input to the sampling approach are the global x, y, z coordinates of all points, and the number of sampled points required. The first step of the sampling procedure is to select a random point and compute the distance from that point to the rest of the points in the segment. The first point is added to the set of visited points. Thereafter, iteratively the algorithm selects the point that has the biggest distance between its previous point and the rest of the unvisited points and adds it to the list of visited points. This continues until the stopping criteria is reached, i.e the cardinality of the visited set is equal to the input parameter. In this manner the selected points that we convolve are separated as far apart as possible. This helps the convolution retain similar amount of information but with only a small fraction of the points.

This algorithm has been efficiently parallelized using CUDA on a GPU (23). Furthermore, the authors present results on the effect of randomness when choosing the first point. When training our model we still use the GPU as much as possible as this computation happens offline

but for test time we resort to our efficient CPU implementation, which we detail below.

In our work, we wish to utilize sampling in each layer of our architecture while being able to run on a CPU. Initially, we developed our approach to iteratively choose the most distant point in each segment individually. This per-segment approach is not optimal, as it requires $O(PS)$ time complexity, where P denotes the number of segments in a batch, while S – the number of selected points to sample. We improved this by implementing a sampling technique directly on batches of segments in a tensor simultaneously. The data in each layer of our model is structured as a tensor, where the height of the tensor holds each point from the segment, the width corresponds to each of the points’ dimensions (usually x, y, z), and the length represents the batch of segments in a point cloud. This way we can efficiently compute the distance matrix of the entire tensor, along the length of the tensor, and simultaneously select the farthest point in all segments.

While the aforementioned approach still needs to compute the distance matrix for each following farthest point, it is still more efficient than running on per-segment basis – $O(S)$. In comparison to a per-segment approach, we gain a ≈ 80 x speed-up, while compared to the thousands of cores that CUDA utilizes, we are just ≈ 30 x slower, but only using a CPU. By paying attention to the computational details of the sampling method we allow our network to process an entire point cloud in ≈ 200 ms on a CPU.

Datasets, Robots and Sensors

A special effort has been made to present broad experimental results for our new method as well as other state-of-the-art segment-based localization methods (12, 13, 19, 20). To provide the basis of our analysis we used six different datasets, collected using five different platforms in urban, industrial and natural environments. The platforms were varied - using three kinds of 3D LIDARs and two kinds of 2D push-broom LIDARs. Nearly 320 kilometers of data has been

analyzed in the Results section. We believe that this level of testing is necessary to properly measure and analyze these methods.

Of particular note is our large-scale analysis of the RobotCar dataset (14). As part of this paper we make available an improved version of the dataset by providing refined ground truth poses of its 300 kilometers as a contribution of this paper.

Datasets

The six datasets were collected in urban, industrial and natural environments. A summary of each dataset can be found in Table 4 and a top-down view is shown in Figure 1. A video preview of our algorithm running on the datasets is available in Movie S2 in the Supplementary Materials.

Oxford RobotCar Dataset

The Oxford RobotCar Dataset contains over 100 sequences of a single consistent route through Oxford, UK, captured over a period of a year on a mid-sized car equipped with various navigation sensors including a push-broom LIDAR. The dataset contains many different combinations of weather, traffic and pedestrians. Data was collected in all weather conditions, including heavy rain, night time, direct sunlight and snow. Road and building works over the period of a year significantly changed sections of the route from the beginning to the end of data collection.

Table 4. Description of datasets used in our experiments.

	Oxford RobotCar	KITTI Seq (00,05,06)	Fire Service College (FSC)	George Square Park (GS)	Cornbury Forest (CP)	Burgess Field (BF)
Environment	Urban	Urban	Industrial	Natural, City Park	Natural, Forest	Natural, Forest/Meadow
Dynamics	Moving cars, bicycles, pedestrians	Moving cars, bicycles, pedestrians	Some moving people	Some moving people	None	Some moving people and dogs
LIDAR	SICK LMS-151 @ 50Hz Push-broom	3D HDL-64E @ 10Hz	3D VLP-16 @ 10Hz	SICK LMS-511 @ 75Hz Push-broom	3D HDL-32E @ 10Hz	(train) 3D VLP-16 @ 10Hz , (test) SICK LMS-151 @ 50Hz Push-broom
Platform	Nissan Leaf	Volkswagen Passat	ANYmal legged robot	Clearpath Husky robot	Bowler Wildcat	Clearpath Husky robot
Path Type	On streets	On streets	Path in Oil Rig	Concrete trail	Off-road dirt track	Off-road dirt track
Scene Area	~10x10 km	~450x300 m	~50x50 m	~500x80 m	~1 km x 80 m	~1x0.5 km
Traveled distance	~300 km	~6.5 km	~250 m	~1 km	~2 km	~5.6 km
# of Clouds	1118	134	169	30	1135	130
# of Points	~83000/cloud	~45000/cloud	~11150/cloud	~845000/cloud	~75000/cloud	(train) ~42000/cloud (test) ~210000/cloud

Ground Truth Annotation The existing Oxford RobotCar Dataset provides position estimates using a Global Positioning System (GPS) and Inertial Navigation System (INS) which can be used as a proxy for ground truth. However, for deep learning the object instances in the LIDAR data need to be tightly associated to one another. The error due to GPS is in many cases too large to be satisfactory for this. For this reason we employed two months of work annotating a more accurate ground truth. This improved ground truth allowed us to establish correspondences between LIDAR clouds to the accuracy needed for supervised learning. Two examples scenes from the aligned dataset are shown in Figure 1. For a full preview of the improved dataset, please see Movie S1 in the Supplementary Materials.

The ground truth was created as follows: visual odometry provided a backbone with visual loop closures established within and across the 100 sequences (24). This created an initial map that we then manually corrected by verifying LIDAR consistency and adding individual loop closures where necessary. This provides several samples where each object is seen and allows the feature learning to adapt to viewpoint changes and environment alterations.

Data Organization In this work we have trained all models with 26 of the sequences, captured between June 2014 and October 2015, containing a total of 140,287 samples from 59,503 classes. For testing we have used four sequences from November 2015 that were not used during training. We have established 101,502 matching and 2,208,008 non-matching segments. Two segments were considered a match if the distance between their centroids is less than 0.5 m; and a non-match when the distance is larger than 20 m. For localization, we generated 3D swathes from the push-broom sensor by incorporating the motion of the vehicle for every 30 m of distance traveled. This resulted in a total of 1,118 point clouds across all four testing sequences.

We make the ground truth publicly available as a contribution of this paper. It has been

added to the same web site as the original dataset[†]. We believe this is very valuable as such a detailed annotation is difficult to attain. We hope it can be useful to other deep learning techniques such as feature learning and change detection.

KITTI

The KITTI dataset (5) provides benchmarks for stereo matching, optical flow, visual odometry, 3D object detection and 3D tracking. The sequences used in this paper are the LIDAR Sequences 00, 05 and 06. We have specifically used the processed KITTI data used in (19, 20), we have done so to facilitate comparison with state of the art methods in a fair way. The sequences were captured by driving around the city of Karlsruhe, Germany, in rural areas and on highways. Up to 15 cars and 30 pedestrians are present at any time in a sequence. Similarly to our baselines, we have used Sequences 05 and 06 for training and Sequence 00 for testing. Specifically, there are 44,802 testing matches and non-matches. Further details in Table 4 and visualizations in Figure 1.

Fire Service College

The Fire Service College is a former Royal Air Force base converted to a fire-fighter training facility. One of its test sites is a mock-up of an off-shore oil rig. It has many bare metal grid surfaces that introduce challenges for LIDAR localization. In this environment the ANYmal quadruped robot (25) was tested carrying out industrial inspection tasks and moving around the facility. The robot was equipped with a Velodyne VLP-16 LIDAR sensor.

George Square

The George Square dataset consists of well-observed and spatially separated trees, benches and bushes in parkland. The experiment was performed with a Clearpath Husky robot traversing

[†]<https://robotcar-dataset.robots.ox.ac.uk>

two loops of ≈ 500 m each. The first loop was used to create the map, while the second test the localization approach. The vehicle was equipped with a push-broom LIDAR. Visual odometry (26), was used to create 3D LIDAR swathes for both the prior map and the scene clouds. The prior map was further corrected in a SLAM system by manual loop closure detection and optimization (27). For the scene clouds, LIDAR swathes were accumulated every ≈ 10 m that the vehicle traveled.

Cornbury Forest

This dataset demonstrates the potential of segment-based localization in a thick forest. The data for the experiment was collected in Cornbury Forest, Oxfordshire, while traversing an approximately 1 km path with a Bowler Wildcat equipped with a Velodyne HDL-32E LIDAR. A picture from a camera mounted on the robot in this environment is shown in Figure 1. This environment is very challenging to localize in, even for a person to manually compare the prior map to a scene point cloud. It consists of heavily interleaved foliage and trees and bushes without clear spatial separation.

The map was created by combining an initial vehicle trajectory, made by registering scans spaced every 30 m to one another using Iterative Closest Point (28), with carefully chosen and reliable GPS measurements. To do this we smoothed the initial trajectory with the GPS measurements using iSAM (27) to achieve a better estimate of ground truth. Test point clouds were created when traversing the environment in the opposite direction by accumulating point clouds for every ≈ 1 m traveled.

Burgess Field

Burgess Field is a dataset collected in a natural environment consisting of a forest and a meadow located near Oxford. The dataset was captured by a Clearpath Husky UGV equipped with a forward facing Bumblebee 3 stereo camera, a Velodyne VLP-16 LIDAR and a push-broom

SICK LMS151 LIDAR. The vehicle traversed a loop of ≈ 2.8 km several times. We used the LIDAR data of Sequence 02 for training and map creation, and the push-broom data from Sequence 01 for testing the classifiers. These two sequences were collected two weeks apart each other in February when the trees had no foliage. The prior map from the push-broom LIDAR of Sequence 02 was built using the method proposed in (24). The test swathes were built using the same visual odometry method for every 22 meters the vehicle traversed in Sequence 01.

To label the data for training, four consecutive Velodyne scans are combined using the motion estimator from (24) and segments are extracted from them. All segments within 0.5 m in consecutive point clouds are considered the same object and labeled as such. All segments further than 4.0 m apart are considered a non-match. In this manner we extracted a total of 82,871 segments across 38,714 classes for training and 45,409 segments in 20,583 classes for testing. For our localization experiment the prior map consisted of 2,783 segments — individual trees, bushes, and interleaved shrubs. Individual scene clouds were created for every 22 m traveled with each scene cloud containing an average of 46 segments. Summary details for the dataset are shown in Table 4 with visualizations in Figure 1.

Robotic Platforms

The different LIDAR localization methods have been tested on a wide variety of robotic platforms. The platforms were: a Nissan Leaf for the Oxford RobotCar dataset, a Volkswagen Passat used in the KITTI dataset, an ANYmal quadruped robot used in the Fire Service College dataset, a Clearpath Husky robot used in the George Square and the Burgess Field datasets, and lastly the Bowler Wildcat used in the Cornbury Forest dataset. Visualizations of the vehicles are shown in Figure 1.

The use of each platform is in accordance to the type of terrain that needs to be tackled.

For urban areas, the cars (Leaf and Passat) are the most suitable choice. The ANYmal robot is used in the Fire Service College due to its ability to walk on stairs and go over obstacles which only a legged robot could tackle. The Bowler Wildcat is deployed in an off-road track in a forest. In the parks and meadows we deployed a Husky. It is important to note that the variety of platforms offer different traversal speeds. Together with the type of sensor this influences the generated scene and map point clouds.

LIDAR Sensors

The utilized LIDAR sensors belong to two different families of sensors - 2D single beam and 3D multi beam. The different types of LIDAR sensors are employed to show that the presented methods are not limited to training and testing on the same sensor.

The 3D sensors are Velodyne VLP-16, HDL-32E and HDL-64E LIDARs, with the number indicating the number of scan beams. In the more dense LIDARs (32 and 64 beams) there is sufficient data present to perform localization with each point cloud but for the 16 beam model we accumulated scans to create denser point clouds. The second family of sensors are the 2D LIDARs, in our case we use SICK LIDARs, the LMS 151 and the LMS-511 which have different scanning ranges, densities and angular resolution. 2D sensors scans need to be accumulated using odometry before processing, this creates less candidate point clouds for localization but typical point clouds are more detailed and less noisy than the 3D sensors. The location of the LIDARs used for localization are marked with a red circle in Figure 1 **D-H**.

References

1. M. Miller, S.-J. Chung, S. Hutchinson, *Intl. J. of Robotics Research* **37**, 13 (2018).
2. L. Paull, M. Seto, J. J. Leonard, H. Li, *Intl. J. of Robotics Research* **37**, 21 (2018).

3. H. Goforth, S. Lucey, *IEEE Intl. Conf. on Robotics and Automation (ICRA)* (2019), pp. 2974–2980.
4. J. Martnez-Gmez, I. Garca-Varea, M. Cazorla, V. Morell, *Intl. J. of Robotics Research* **34**, 1681 (2015).
5. A. Geiger, P. Lenz, C. Stiller, R. Urtasun, *Intl. J. of Robotics Research* **32**, 1231 (2013).
6. J.-L. Blanco-Claraco, F. ngel Moreno-Dueas, J. Gonzlez-Jimnez, *The International Journal of Robotics Research* **33**, 207 (2014).
7. T. Pire, M. Mujica, J. Civera, E. Kofman, *The International Journal of Robotics Research* **38**, 633 (2019).
8. J. P. Underwood, G. Jagbrant, J. I. Nieto, S. Sukkarieh, *J. of Field Robotics* **32**, 1056 (2015).
9. J. Garforth, B. Webb, *IEEE Intl. Conf. on Robotics and Automation (ICRA)* (2019), pp. 1794–1800.
10. F. Ma, L. Carlone, U. Ayaz, S. Karaman, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)* (2016), pp. 96–103.
11. J. L. S. Rincon, P. Tokekar, V. Kumar, S. Carpin, *IEEE Robotics and Automation Letters* **2**, 2016 (2017).
12. G. Tinchev, S. Nobili, M. Fallon, *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)* (2018), pp. 8239–8246.
13. G. Tinchev, A. Penate-Sanchez, M. Fallon, *IEEE Intl. Conf. on Robotics and Automation (ICRA)* (2019), pp. 1327–1334.
14. W. Maddern, G. Pascoe, C. Linegar, P. Newman, *Intl. J. of Robotics Research* **36**, 3 (2017).

15. A. Krizhevsky, I. Sutskever, G. E. Hinton, *Conf. on Neural Information Processing Systems (NIPS)*, F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger, eds. (Curran Associates, Inc., 2012), pp. 1097–1105.
16. J. Deng, *et al.*, *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition* (Ieee, 2009), pp. 248–255.
17. Z. Wu, *et al.*, *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition* (2015), pp. 1912–1920.
18. A. Kuznetsova, *et al.*, *Intl. J. of Computer Vision* (2020).
19. R. Dubé, *et al.*, *IEEE Intl. Conf. on Robotics and Automation (ICRA)* (2017), pp. 5266–5272.
20. R. Dubé, *et al.*, *Robotics: Science and Systems (RSS)* (Pittsburgh, Pennsylvania, 2018), pp. 20–30.
21. O. Chum, J. Matas, *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition* (2005), pp. 220–226.
22. Y. Li, R. Bu, M. Sun, B. Chen, *arXiv preprint arXiv:1801.07791* (2018).
23. C. R. Qi, L. Yi, H. Su, L. J. Guibas, *Conf. on Neural Information Processing Systems (NIPS)* (2017), pp. 5099–5108.
24. W. Churchill, P. Newman, *Intl. J. of Robotics Research* (2013).
25. M. Hutter, *et al.*, *Advanced Robotics* **31**, 918 (2017).
26. A. S. Huang, *et al.*, *Intl. J. of Robotics Research* (Springer, 2017), pp. 235–252.

27. M. Kaess, A. Ranganathan, F. Dellaert, *IEEE Trans. Robotics* **24**, 1365 (2008).
28. F. Pomerleau, F. Colas, R. Siegwart, S. Magnenat, *Autonomous Robots* **34**, 133 (2013).

Acknowledgments

Funding: This work was supported by EPSRC RAIN and ORCA Robotics Hubs (EP/R026084/1 and EP/R026173/1 respectively). M. Fallon is supported by a Royal Society University Research Fellowship.

Supplementary materials

Materials and Methods

Movie S1. Preview of the aligned RobotCar dataset.

Results

Movie S2. Preview of the algorithm operating in urban, forest, and industrial scenes.

Summary

Following the progress we made with ESM in chapter 4, the natural next step would have been to explore the generalization abilities of this system by testing and training in different environments. This chapter performed thorough ablation studies of the proposed systems not only in terms of overall registration quantity, but also tested features trained in different environments. This allowed us to determine the bottlenecks of the system and we proposed a new compact architecture that provided further memory and computational cost reductions.

One key discovery was that point-based networks' main computation bottleneck was in the number of points processed. This gave us incentive to apply a sub-sampling strategy that would reduce the number of points processed per layer, without sacrificing performance. In our empirical findings, it was crucial to devise a strategy for hierarchical feature aggregation based on coverage. Another empirical observation from this chapter is the fact that point-based networks are not invariant to rotations. We tackled the issue by introducing a pre-processing alignment step prior the description stage of the segments. While this did not provide full rotation invariance it was able to alleviate the issue significantly. Another interesting finding is the ability of deep learning to generalize to unseen challenging data with minimal loss in performance.

Due to the nature of the proposed algorithm, ablation studies required large scale experiments. Therefore, this chapter also saw the need for substantial data and presented an extension of the RobotCar dataset [Maddern et al. 2017]. The processed data provided per-point annotation for 30 sequences of the original dataset that can be used for supervised training of deep learning models.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Efficient LIDAR-based Global Localization
Publication Status	☐ Published ☐ Accepted for Publication ☒ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Tinchev, G., Penate-Sanchez, A., & Fallon, M. (2021). Efficient LIDAR-based Global Localization. <i>IEEE International Conference on Robotics and Automation (ICRA)</i>

Student Confirmation

Student Name:	Georgi Tinchev		
Contribution to the Paper	- Primary author of the work - Designed experiments - Evaluated the network against state-of-the-art related research methodologies - Conducted detailed ablation studies - Analysed rotation invariant modules within the network - Post-processed the Oxford Robotcar dataset to obtain accurate alignment - Studied the generalization ability of the proposed approach		
		Date	30/12/2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Dr. Maurice Fallon		
Supervisor comments: The contributions listed above are accurate		
	Date	30/12/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

6

SKD: Keypoint Detection for Point Clouds using Saliency Estimation

To this end, chapters 3 to 5 advanced feature learning in the context of place recognition while also designing efficient deep learning models with optimal performance. However, a constant part of the segment-based pipeline remained the segmentation method which still needs tuning for each specific environment. In the following chapter we propose an alternative to the engineered segmentation method that directly learns interesting keypoints in the environment. The points surrounding the keypoints can then be considered as a segment and used for place recognition. A key objective is that the proposed method should be applicable to any differentiable descriptor, such as the ones used in chapters 4 and 5, without the need of retraining for specific environment. In the following we present an approach for extracting keypoints based on saliency estimation, named Saliency Keypoint Detector (SKD).

This work was submitted to the IEEE Robotics and Automation Letters, 2021 with a presentation option at the IEEE International Conference on Robotics and Automation, 2021.

SKD: Keypoint Detection for Point Clouds using Saliency Estimation

Georgi Tinchev, Adrian Penate-Sanchez and Maurice Fallon

Abstract—In this work we present a novel keypoint detector that uses saliency to determine the best candidates from a point cloud for tasks such as registration and reconstruction. The approach can be applied to any differentiable deep learning descriptor by using the gradients of that descriptor with respect to the input to estimate an initial set of candidate keypoints. By using a neural network over the set of candidates we can also learn to refine the point selection until a final set of keypoints are obtained. The key intuition behind this approach is that keypoints are not extracted solely on the geometry surrounding a point, but also take into account the descriptor’s response to a given task. To improve the performance of the learned keypoint descriptor we combine the saliency, the feature signal, and geometric information from the point cloud to allow the network to select good keypoint candidates. The approach was evaluated on two large LIDAR datasets - the Oxford RobotCar dataset and the KITTI dataset, where we obtain up to 50% improvement over the state-of-the-art in both matchability score and repeatability. When performing coarse registration with the keypoints suggested by our method we achieve a higher inlier ratio and faster convergence.

I. INTRODUCTION

A key task in robotic localization or reconstruction pipeline is the repeatable extraction of points that can be reliably matched with a map or representation of an environment or object. Image keypoint extraction has been well studied, but point cloud keypoint extraction is less explored. In this work, we present a novel method for keypoint extraction from point clouds that exploits the spatial information encoded around each feature and proposes points with higher likelihood of being matched between two point cloud observations. The concept of jointly learning the *keypoint and feature* pair leads to performance improvement.

Traditionally, keypoint extraction methods have only focused on the local geometry of the image, such as the Harris corner detector [1] or the more efficient Difference-of-Gaussians approach which is used in SIFT [2]. These methods focused on understanding the geometry of the image so as to match points present in a pair of images, known as *repeatability*. If corresponding points are obtained from two different images of the same scene, one can then find matches between them without introducing incorrect matches (outliers). If the outlier percentage is low, a robust estimator [3] can converge faster and produce a better solution. The problem with this approach is that it ignores how the

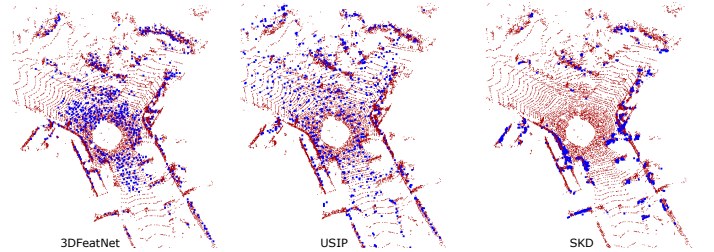


Fig. 1. Top-down view of a single point cloud (brown) and 1024 generated keypoints (blue). Our method (SKD) was not implicitly trained to ignore the ground, but rather it uses the feature signal of the descriptor and the context information to determine informative areas in the environment - corners, edges and structure.

feature will react to disturbances. The fact that a point is in the same physical place does not mean that the descriptor will produce the same response. Therefore, one has to look at the combination of *keypoint and feature* performance. These are the two factors in feature matching, thus in this paper we focus on namely *matchability*. As evidenced by recent research [4], repeatability is useful but not sufficient for evaluating keypoint quality. The only reliable metric is matchability, as it measures how useful the combination of *keypoint and feature* is. For this reason we focus on building a keypoint detector that learns to produce points that can be reliably matched given a deep learning descriptor. All our experiments are based on the state-of-the-art feature descriptor for point cloud data — 3DFeatNet [5]. We use the gradient response of the descriptor with respect to the input point cloud and combine it with contextual information from the point cloud in order to produce keypoints that generate a higher percentage of inlier matches.

The proposed approach leverages the idea, presented in [4], that state-of-the-art keypoint detectors can be built by using only the feature gradients with respect to the input. The gradients encode much of the local information used by the feature descriptor. The points of maximum gradient response provide a set of very promising candidate points. Using this initial set of keypoints, we enforce the notion of *saliency* that uses the combination of neural network layer activations and the gradients at that layer w.r.t. the input to select more characteristic keypoints (Fig. 1). The use of saliency has been shown to produce promising results both in images and in point clouds [6], [7], [8]. This subset of the original points can then be combined within a neural network that looks at the geometry of the point cloud and makes a decision as to which are the best points to select. The main contributions of our work are as follow:

- Saliency-based keypoint extraction: a novel method for

The authors are with the Oxford Robotics Institute, University of Oxford, United Kingdom. {gtinchev, adrian, mfallon}@robots.ox.ac.uk

This work was supported by EPSRC RAIN and ORCA Robotics Hubs (EP/R026084/1 and EP/R026173/1 respectively) and EU H2020 project Memory of Motion (MEMMO, project ID: 780684). M. Fallon is supported by a Royal Society University Research Fellowship.

extracting keypoints in point clouds based on the gradient response of a differentiable descriptor, the context information, and the geometry of the scene.

- A context agnostic approach: by combining the gradient response with the semantic information of a point cloud, our approach is less vulnerable to biases in the training data. This makes our approach robust when testing on a different context and allows the model to be used without retraining. We demonstrate this by training on the Oxford RobotCar dataset [9] and testing on the KITTI dataset [10] without retraining.
- State-of-the-art performance: Compared to the state-of-the-art [5], [11], our approach generates two times more correct matches and achieves 40% more relative repeatability. We have evaluated our algorithm on two of the largest and most widely used LIDAR datasets - the Oxford RobotCar dataset [9] and the KITTI dataset [10]. In total these datasets contain point clouds collected from more than 300 kilometers of experiments.

II. RELATED WORK

This paper focuses on reliably obtaining keypoints from point clouds. This is a topic which is very closely related to image keypoint extraction, so we will review the literature for both fields in this section. We will also outline the relevant research on point cloud-based deep learning models and saliency methods as these topics are central to the contributions of our approach.

A. Image Keypoint Detectors

Much of the recent work on keypoint and feature extraction from images employs deep learning architectures. In [12] a keypoint detector for depth images was presented. It used a Siamese approach by pairing two Faster-RCNN networks [13] with a contrastive loss between them. Quad-Networks [14] are an example of unsupervised keypoint generation that learned to rank points. The authors kept a ranking before and after the transformation while looking at the top and bottom quartiles of the distribution of keypoints. Methods such as [15] and [16] extract unsupervised landmarks. They find the best possible cues to improve the performance of a given task.

Approaching landmark generation in an unsupervised fashion is the equivalent to learning keypoints, because no ground-truth points exist and thus the regions that maximize the end-task performance must be manually selected. Specific to unsupervised keypoint learning, [17] presented an approach that generated reliable keypoints by learning from the temporal consistency of the network activations in short videos. A successful approach to obtaining good keypoints is to optimize how they respond to the content of the image and not just the geometry. TILDE [18] learned to make the points more reliable by understanding how changes in weather and lighting conditions modified the performance of a point. In contrast, our work focuses not only on the reliable geometry, but also looks at the content and the context of a point.

In LIFT [19], the keypoint detector and the feature descriptor are jointly learned by leveraging structure from motion sequences to generate large amounts of training data. By understanding that the job of a keypoint is to increase the probability of descriptor matching, LIFT focused on matching performance and at the time outperformed all previous approaches. Very recently, ELF [4] applied a simple approach by using the gradient response of deep learning features to produce keypoints. They managed to outperform both TILDE and LIFT without needing to train. Their approach obtained very reliable embedding information by looking at the feature activations.

B. Point Cloud Keypoint Detectors

A series of approaches built upon PointNet [20] and PointNet++ [21] introduced new ways to efficiently understand unstructured sets of points (point clouds). By learning a symmetry function approximated by a multi-layer perceptron (MLP) the authors proposed a new kind of layer that could learn an approximation of a convolutional operator over sets of unordered points. Similarly, [22] managed to increase the performance of the basic neurons by approximating a convolution using an MLP and by performing aggregation of spatial data within each neuron. [23] presented a method to perform place recognition from point clouds. The point clouds were described using a combination of networks trained with a metric learning loss to produce a feature vector.

Due to the success of the aforementioned approaches, many novel methods leveraged PointNet layers in their work [24], [8], [25]. Recently, in [24], the authors studied how adversarial attacks could affect the performance of the PointNet layers. It also studied how introducing or removing points affected performance. In [8] the authors built a saliency map to understand the effect of each point on the final prediction. They tested the performance of their saliency score by performing point dropping operations to demonstrate performance better than a method based on the critical-subset theory. Frustum PointNets [26] used PointNet layers as the building blocks for their approach and applied it to object detection in point clouds. Their approach produced good quality performance by combining image inputs with point cloud inputs. Also combining images and point clouds for 3D object detection, PointFusion [25] presented a method that focused on bounding box prediction by proposing a novel dense fusion architecture.

The most relevant methods to our work are 3DFeatNet [5] and USIP [11]. 3DFeatNet [5] is the state-of-the-art in feature extraction for point clouds. It uses PointNet++ as a building block and learned a detector and a descriptor by using a two stage network. The main problem with this approach is that the keypoint extraction network does not perform as well as desired. Therefore, recently a novel keypoint extraction method was proposed, USIP [11]. It focused on obtaining keypoints with a high level of repeatability. This increases the performance of the USIP detector compared to the 3DFeatNet detector significantly.

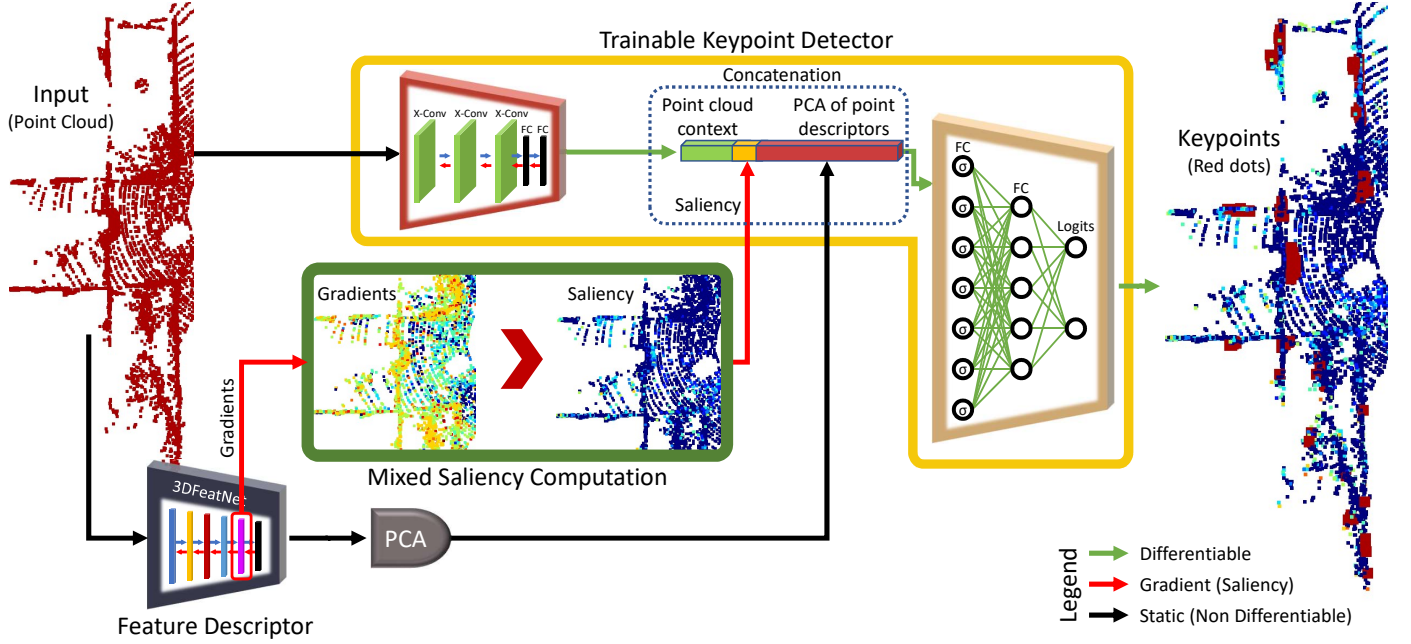


Fig. 2. Proposed network architecture. The input to our system is a raw point cloud and a feature descriptor network. Our method concatenates the *saliency* information from the gradients of the feature descriptor with the per-point context features and a PCA projection of the original feature descriptor. The concatenated vector is the input to two FC layers that generate per-point response at the logits layer.

C. Saliency Estimation

Saliency has been studied as a way to quantify and understand what is relevant within a neural network. The definition of *saliency* in this context has been studied by a number of works, [6], [27], [28] among others, that seek to understand why machine learning models behave as they do. Even before the growth in popularity of neural networks, methodologies such as [29] were being designed to understand why classifiers made specific decisions. This became more prevalent with the adoption of deep learning models for most perception tasks. Approaches such as [6], [28] seek to understand what a neural network finds relevant by looking at how the gradients of a given prediction behave. The interpretation was that gradients with higher magnitude in specific areas influence predictions. The use of *saliency* has been demonstrated to be capable of generating state-of-the-art keypoint extractors in images. ELF [4] computed the gradient of the feature map given an image then used the Kapur threshold [30] to select keypoints. In a similar fashion, Grad-CAM [27] used the gradient maps of a classification score to produce regions of interest for a given image that can aid tasks such as classification, image captioning, and visual question answering.

III. METHODOLOGY

We present Salient Keypoint Detection (SKD) - a keypoint extraction method based on the *saliency* of a point cloud. We define *saliency* as the combination of the feature activation signal at a specific layer of a pre-trained descriptor and the gradients of the same layer with respect to the input point cloud. We leverage a trained descriptor network to produce robust keypoints directly from 3D data. We then project the *saliency* into the spherical coordinate system of the input

point cloud and extract the most informative regions. We combine those regions with *context-aware* features and the features of the original descriptor in order to extract robust and repeatable keypoints. Using the combined feature vector, we train a neural network that learns to predict the likelihood of a point being a keypoint. Similarly to the approach of [4] on images, we extract the gradients of the features at different levels of the architecture, and choose the best performing one. An evaluation of the performance at each layer is presented in Sec. IV-C. In contrast with [4], we select keypoints by combining the gradient information and the activations with the points in Euclidean space in order to determine critical points using the criterion defined by [8].

A. Point Cloud Saliency

For a given point cloud $P \in \mathbb{R}^3$, we extract the gradients of a pre-trained network ∇F at a specific layer l with respect to the input, defined as ∇F_l^P . We define the *initial saliency* $S(P)$, as the product of the feature activations of that layer F_l^P with the gradients, formally defined as:

$$S(P) = F_l^P \cdot \nabla F_l^P \quad (1)$$

In this way the extracted initial saliency corresponds to specific points in the point cloud that have good activations and are valuable based on the gradient of that layer w.r.t. the input. From a geometric perspective this can be thought of projecting the feature signal through to the input point cloud, determining the value of individual points. An example projection can be seen in Fig. 3. The initial saliences are of the same dimension as the input point cloud, $S(P) \in \mathbb{R}^3$.

We take these saliences and determine a secondary saliency score per point, defined as s_i for each point $i \in P$. The score weights the contribution of $S(P)$ by the distance

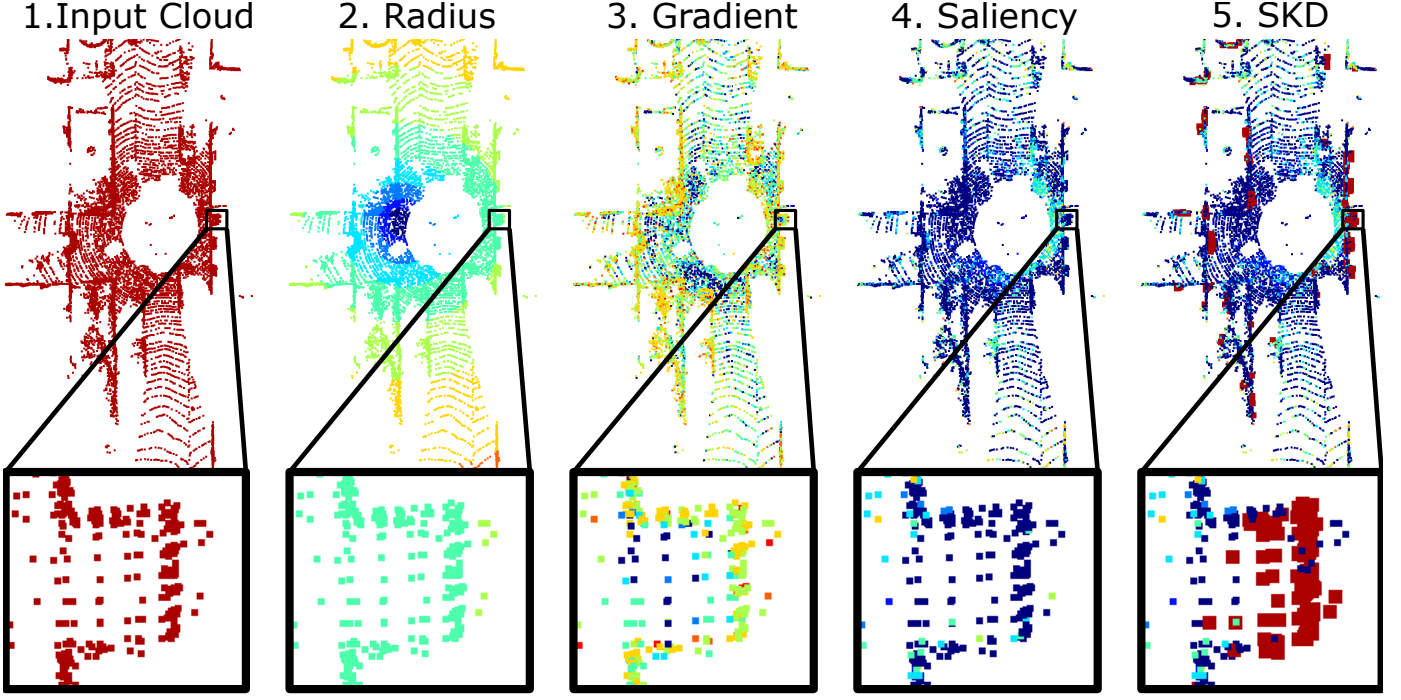


Fig. 3. **1.** Input point cloud P . **2.** Distances of each point (r_i) from the spherical core - blue indicates closer to the median point, red - further apart. **3.** The extracted $S(P)$ signal based on the activations and gradients w.r.t. input point cloud (1.) - blue corresponds to negative gradients, red - highly positive. **4.** The *initial saliency* score s_i estimated from the radius (2.) and the gradients (3.), blue - low score, red - high score. **5.** Selected keypoints (red, enlarged) on top of the saliency score. Our method picks keypoints based on the keypoint saliency score s_i , but also leverages context and the original descriptor to correctly pick out edges and corners negatively scored by s_i .

from the center of the point cloud in the Spherical Coordinate system. The score is formally defined as:

$$s_i = - \sum_{j=1}^3 [S_j(P) \odot (x_{ij} - \text{median}(x_{ij}))] r_i \quad (2)$$

where $j \in \{1, 2, 3\}$ defines each of the Cartesian coordinates x, y, z of point $i \in P$, $\odot$ is the Hadamard Product and $r_i = \sqrt{\sum_{j=1}^3 (x_{ij} - \text{median}(x_{ij}))^2}$ is the distance of point i to the median of the spherical core of the point cloud [8]. In other words, we transfer the gradient and its activations under the orthogonal coordinates and measure the offset from the center of the point cloud.

This transforms the original point cloud to a spherical coordinate system. Further away points will have higher score, and points closer to the centre will have lower score (see Fig. 3). The reason for doing this is that under any rotation and translation of the input cloud, the centre of the point cloud will not change, and is therefore less informative. Furthermore, when performing geometric registration, selected models (points) that are further apart will provide a more robust solution in comparison to points which are very close together.

Finally, the saliences are normalized to have zero mean and unit variance within a single point cloud. The final saliency ensures a good spatial distribution of the selected points while also selecting points with good activations based on the descriptor network. Fig. 3 illustrates the saliences of an example input point cloud at each stage of computation.

B. Network Architecture

Our network architecture is depicted in Fig. 2. It consists of three parts. The first part is called the *point cloud context* features. This is an ensemble of four X-Conv layers [22] and two fully connected layers that have been pretrained on a feature extraction task to create stable initial estimates. These layers will learn to provide a description of the context around any given point. We use a two dimensional size for the context latent space. The second component is the *saliency* method described in the previous section. The final component is a PCA dimensionality reduction of the original per-point features which improved the final results by producing a smoother feature space. The three components are concatenated and fed to two additional fully connected layers to produce the final keypoint prediction. The network learns to infer a score for each point — determining the probability of it being a robust and repeatable keypoint for the original descriptor. Note that our model is descriptor-agnostic and thus can be applied to any descriptor network in order to improve the performance.

1) Training: During training, the input to our model consists of a tuple of point clouds and the ground truth transformation between them, namely (P_k, P_l, T) . Both P_k and P_l are $N \times 3$ dimensional vectors, where N denotes the cardinality of the point cloud set. In addition, we assume to have a pre-trained model for the point descriptors. To achieve this we estimate the saliency s_i and features f_i for each point in both point clouds. Due to the large dimensionality of the feature space f_i , we performed PCA and transformed the

features that explained $\approx 90\%$ of the data. This smooths the feature space, which leads to better results.

In addition, given the ground truth transformation between the two point clouds, we determine the bidirectional correspondence for each point, conditioned on the descriptor. These correspondences are used in our loss to select matching pairs of points. Note that these correspondences need not be injective nor surjective. In other words, if for a point i_1 the closest neighbour under the ground truth transformation, T , is j_1 , it does not mean the reverse applies, nor does that limit i_1 to have a unique neighbour from P_l .

For each point in both point clouds we extract what we have called *context-aware features*, f_c , and concatenate them to the saliencies s_i and features f_i . The term *context-aware* is used as we expect the layers that lead to this descriptor to contain information about the local geometry around each point, helping the network to understand the correlations between the local geometry and the point descriptors. We pretrain these layers on a feature learning task to obtain stable initial features in the training. We chose a small feature space of only two dimensions in order to force the network to learn rough estimates of the shape of objects that can generalize better when moving to a different dataset. Afterwards, the full architecture is trained end-to-end with the saliency and feature concatenation in the middle to create a rich and informative latent space.

The concatenated saliencies, PCA features, and *context-aware* features are fed to two fully connected layers to estimate the probability of each of the points being a keypoint. To this end we use a softmax cross entropy loss between the stacked P_k, P_l clouds and the determined correspondences, given the ground truth transformation. Due to the smaller number of correct keypoint correspondences between the two clouds, we balance the loss function terms given the keypoint to non-keypoint ratio determined by the ground truth correspondences.

2) *Inference*: During the forward pass of the network we estimate a probability for each point being a correct keypoint. We can then either extract the top K keypoints, based on this probability or select all the keypoints based on the probability of each point. The extracted keypoints produced by the described approach are better suited to the descriptor as the learning iterations optimize its performance. Note, that neither Non-Maximum Suppression, nor any other threshold is applied to obtain the final set of keypoints.

IV. RESULTS

In this section we discuss the datasets and metrics we used to evaluate our approach and then present our findings.

A. Datasets

In our study we used two datasets - the Oxford RobotCar [9] and the KITTI odometry dataset [10]. In order to provide a fair comparison, our experiments use the preprocessed test data provided by [5] and [11] and we also use their evaluation scripts to make the comparison as fair as possible. We train our model using the same sequences from

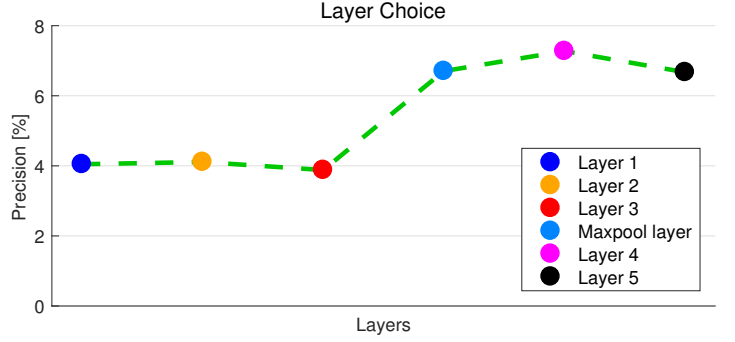


Fig. 4. Contribution of each layer of 3DFeatNet expressed as percentage of correct matches if the layer was selected for gradient computation. The layer identified corresponds to the counter of layers in [5] starting from the left. In our evaluation we used the layer that maximized the performance (Layer 4) for both ELF [4] and our method.

the RobotCar dataset as [5], and test our approach using the same test set of 3,426 point cloud pairs which the authors provided. Furthermore, we do not train our method nor these baselines on the KITTI dataset, in order to test the generalization ability of the proposed approach.

The evaluation part of the KITTI dataset, used by both [5], [11], provides only 2,369 point clouds out of the total dataset. So as to increase the size of the KITTI evaluation dataset, we extended it using the 11 *training* sequences. This is possible only because the RobotCar dataset is used for training all models. The extended dataset is processed in a dense manner: for each point cloud we align the next consecutive 10 point clouds to it using the ground truth transformation. By doing this we expanded the number of testing point cloud pairs from 2,831 to 207,917, which allows us to more fully study our proposed approach.

B. Metrics

We used three metrics to compare the performance of our method. We focused our analysis on the keypoint extraction methods. We use the same 3DFeatNet [5] descriptor for all the methods in order to compare the performance of the keypoint detectors.

The first metric uses the matching score proposed by [5]. We detect keypoints separately for two point clouds. Given the ground truth transformation, we project the keypoints from the first point cloud into the second one. Keypoints that do not have a nearest neighbour correspondence in the second point cloud are ignored from the final result (i.e. no overlap). For the rest of the keypoints, the descriptors are compared and matched to establish a correspondence. The precision is measured as the number of correct correspondences against the total number of possible matches. The metric estimates the percentage of correct correspondences based on the distance between them. Correct correspondences are thresholded to 1m distance, as it serves as a desirable upper limit when performing registration between two point clouds.

For the second metric we chose to compare the normalized relative repeatability, as proposed in [11]. The metric

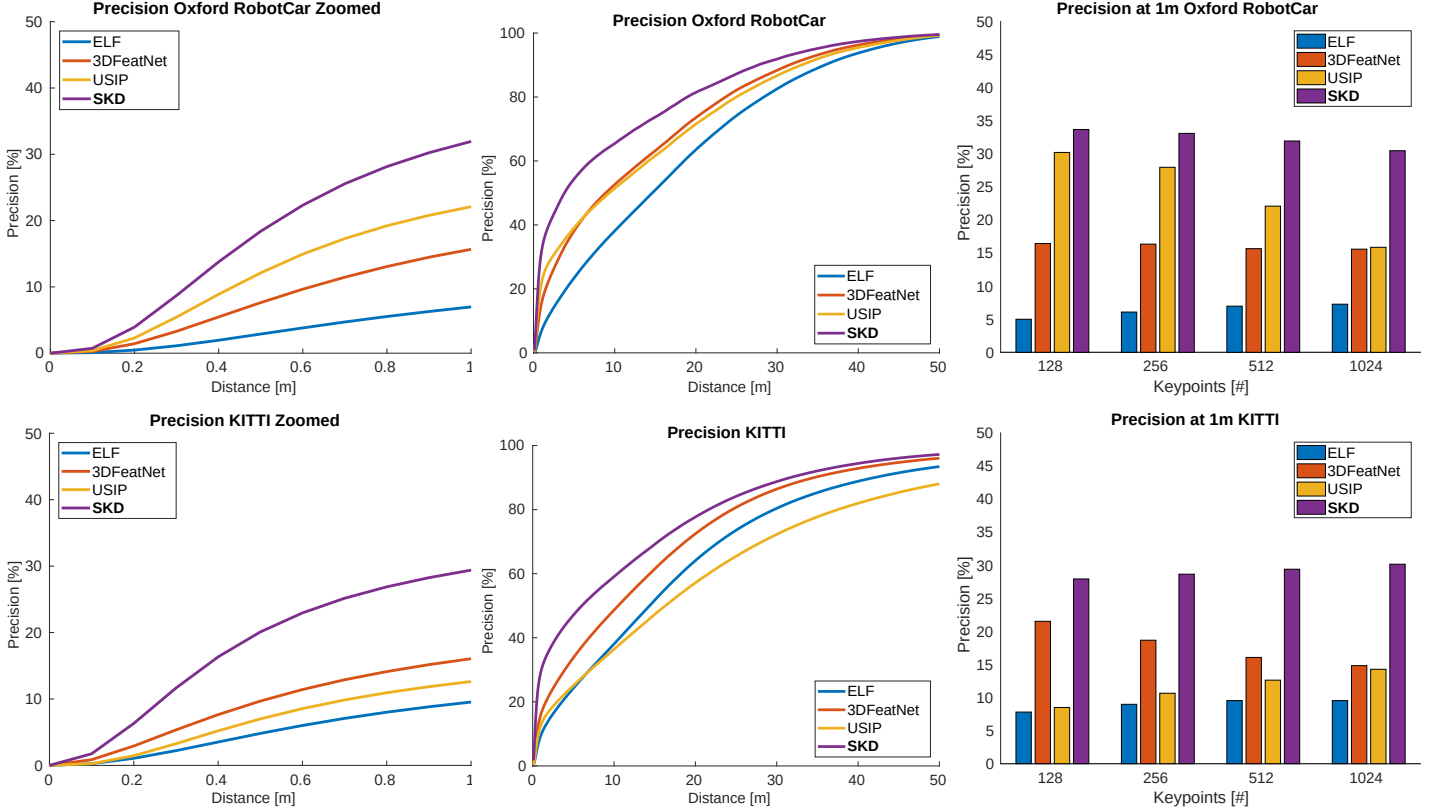


Fig. 5. Matching score evaluated on the Oxford RobotCar dataset (top row) and the KITTI dataset (bottom row). The first two columns are the percentage of matched keypoints using the 3DFeatNet [5] descriptor when varying the distance between correspondences. The first column shows a zoomed-in version of the second column within 1 m of distance, considered relevant for geometric registration. The third column shows the performance of different approaches while varying the number of keypoints.

compares the keypoints detected in one point cloud to the keypoints detected in its corresponding point cloud. If the keypoints overlap, below a certain distance, they are considered a match. While this saturates with a high number of keypoints [4], we chose to use this metric to have a fair comparison against USIP [11]. We note that the ability to match keypoints on their own is not sufficient, hence we consider the matching score to be a more important metric.

Finally, we evaluated the geometric registration performance of our approach on both the KITTI and the Oxford RobotCar datasets using RANSAC [3]. As in [5], we consider a successful alignment to be it differs by less than 2 m and 5° from the ground truth transformation. For the successful registrations we present relative translation error (RTE), relative rotation error (RRE), success rate as a percentage of the successful registrations, average number of iterations it took RANSAC to find a suitable candidate within 99% confidence (capped at 10,000 iterations), and the inlier ratio of how many points were considered when obtaining a correct registration.

C. Baseline Methods

Our approach is general and can be applied to any point cloud descriptor network. For simplicity we chose to use the descriptor of [5] as it is open source and easy to use, as well as being the state-of-the-art in point cloud descriptors. We select the best performing layer from the descriptor to

generate the gradients on which we compute the saliency values, as evidenced in Fig. 4. We have compared against the learned keypoint detector methods of 3DFeatNet [11] and USIP [5], as well as hand-engineered keypoint extraction methods such as SIFT-3D [2], ISS [31], and Harris-3D [1], and a 3D interpretation of the ELF [4] detector. All the learning methods are trained on the Oxford RobotCar dataset and tested on both RobotCar and KITTI data. We have used the models provided online, and trained our own network. For USIP, we took the models provided by the authors which they trained on the Oxford RobotCar dataset. As ELF does not need training, we took the best performing layer, according to Fig. 4. We adapt their approach to work on point cloud data by performing Non-Maximum Suppression in 3D and choosing keypoints based on the Kapur Threshold [30].

D. Matching Score Experiments

We begin by evaluating our approach using the matching score metric on both Oxford RobotCar and KITTI datasets. Fig. 5 illustrates the performance of SKD compared to other state-of-the-art methods. The first row shows the percentage of matched keypoints on the Oxford RobotCar dataset using the 3DFeatNet descriptor for all methods. The first column is a zoomed-in version of the percentage of points detected within 1 m distance that is considered relevant for matching. The last column shows the precision at 1 m distance at different number of keypoints — 128, 256, 512,

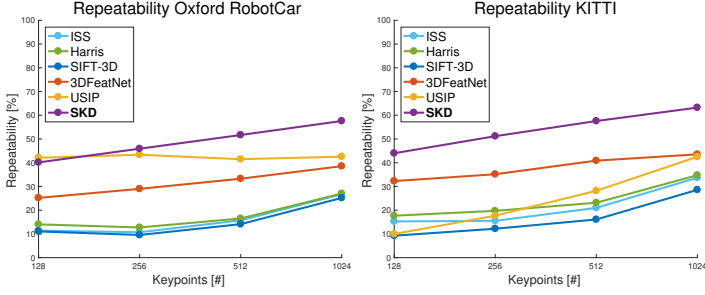


Fig. 6. Quantitative analysis of relative repeatability on the Oxford RobotCar dataset (left) and KITTI (right) measured with different number of extracted keypoints.

1024. As all the approaches are learning-based, we took the top K keypoints. This ensures that the number of keypoints are identical. The second row presents the results for the full KITTI dataset, with all the methods trained on the Oxford RobotCar dataset. We observe that our approach generalizes well without a loss in performance, and outperforms the second best approach by a significant margin. Also, similarly to 3DFeatNet [5], our approach’s performance does not decline significantly when increasing the number of detected keypoints.

E. Repeatability Experiments

Next, we evaluate the repeatability of keypoints across multiple observations of the same scene. Fig. 6 shows the repeatability of SKD measured on the Oxford RobotCar dataset (left) and the KITTI dataset (right). For the Oxford RobotCar, our approach performs comparably to the second best method when extracting 128 and 256 keypoints, but selects points that are 40% more repeatable when extracting 1024 interesting points. Furthermore, Fig. 1, 7, 8, and the supplementary video material present qualitative examples of the detection of keypoints of the top three approaches. The proposed approach has learned to select more descriptive areas of the environment. For example, our method does not select ground points, and even though we forced the method to generate large amounts of points, it still choses points based on the high activations of the network - around building edges and corners, that in turn are easier to match.

Fig. 6 (right) illustrates the repeatability of our approach on the KITTI dataset. On this dataset, 3DFeatNet performs slightly better than on the Oxford RobotCar dataset. Interestingly, as we use the feature signal of 3DFeatNet, our keypoint detector also performs better. In addition, Fig. 7 presents qualitative evaluation of our approach on the KITTI dataset with an increased number of keypoints, where SKD consistently selects the same areas of the environment.

F. Geometric Verification Experiments

Although our work is primarily focused on keypoint detection, we provide results that show one of the possible applications of our keypoint detector — geometric registration. We present the results of the geometric registration on KITTI and Oxford RobotCar datasets in Tab. I, to obtain

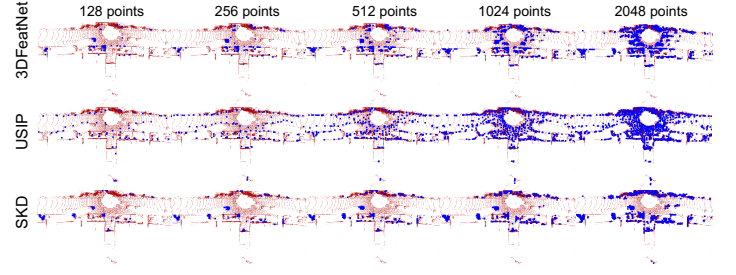


Fig. 7. Qualitative results for the KITTI dataset. For all methods the number of generated keypoints is increased from left to right. This shows what relative importance each method gives to certain areas of the point cloud.

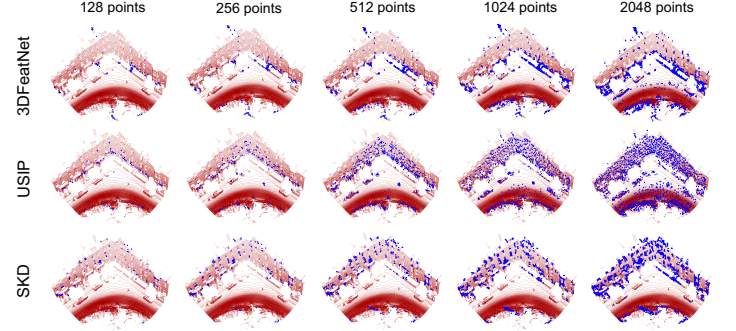


Fig. 8. Qualitative results on the Oxford RobotCar dataset. The keypoint proposals from SKD are more evenly spread in geometrically interesting locations.

this results we have used the code from the authors of the baselines.

SKD performs similarly to the state-of-the-art in terms of relative rotation and translation error, within the standard deviation of the best-performing method (2 mm and 0.04 degrees difference with best baseline). Our method is, however, approximately nine times faster on the KITTI dataset and two times faster on the Oxford RobotCar dataset compared to the second-best approach. This suggests that, although our keypoint detector is not the one producing the rotation and translation estimates, it enables such methods to converge faster due to the higher percentage of inlier points that SKD provides.

G. Ablation studies

Next, we studied the contribution of each component of our method in terms of the precision at 1 m distance on the Oxford RobotCar dataset, shown in Fig. 9.

First, we inspected the contribution of the saliency in the concatenated vector. In our experiments the saliency information alone is responsible for approximately half of the concatenated vector - dark blue compared to yellow in Fig. 9. In addition, training a neural network with just saliency information contributes to approximately a third of the end result - red compared to light blue. For these reasons we have decided to name our method Saliency Keypoint Detector (SKD). Second, the contribution of the network when training with the full feature vector is substantial as is to be expected - purple compared to dark blue and yellow

Method	KITTI					Oxford RobotCar	
	RTE (m)	RRE (deg)	Success Rate	Avg # Iter	Inlier rate	Avg # Iter	Inlier rate
3DFeatNet [5]	0.142 ± 0.120	0.533 ± 0.410	97.80%	3917	12.7%	3083	12.9%
USIP [11]	0.203 ± 0.193	0.673 ± 0.517	97.12%	5324	11.0%	823	28.0%
SKD	0.140 ± 0.134	0.579 ± 0.480	96.52%	594	32.2%	393	32.7%

TABLE I: Geometric registration results on KITTI (left) and Oxford RobotCar (right) datasets with RANSAC. SKD performs commensurately to the state-of-the-art (2 mm and 0.04 degrees difference with best baseline), while achieving a higher inlier rate and requiring fewer number of iterations.

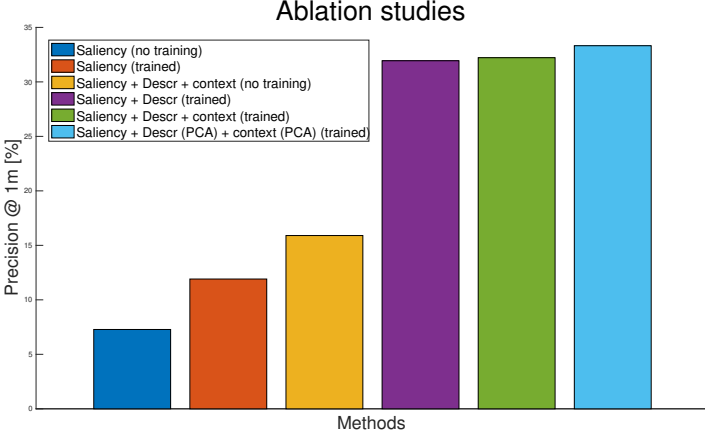


Fig. 9. Ablation studies of our method evaluated on the Oxford RobotCar dataset. We evaluate the contribution of the saliency information, and the importance of training with the original descriptor. The results displayed in the figure show the clear contribution the neural network is making to the keypoint generation process.

on Fig. 9. Third, adding the context features and performing PCA on both the context and the descriptor features slightly increases performance. Our intuition is that PCA smooths the learned feature space, which leads to better generalization.

V. CONCLUSIONS

In this paper we present a novel method for keypoint extraction that uses saliency information to extract informative regions in a point cloud. The method concatenates signals from the gradients with respect to the input, context-aware features, and the descriptor features and learns to predict which 3D points have a higher chance of being matched correctly. The proposed approach is descriptor-agnostic and outperforms the state-of-the-art by up to 50% in matchability and repeatability compared to the second-best method. When performing geometric registration, the algorithm achieves a higher number of inliers while also finding a correct transformation significantly faster than the current state-of-the-art.

REFERENCES

- [1] C. Harris and M. Stephens, “A combined corner and edge detector,” in *Fourth Alvey Vision Conference*, 1988.
- [2] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *IJCV*, 2004.
- [3] M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *CACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [4] A. Benbihi, M. Geist, and C. Pradalier, “ELF: embedded localisation of features in pre-trained CNN,” in *ICCV*, 2019.
- [5] Z. J. Yew and G. H. Lee, “3dfeat-net: Weakly supervised local 3d features for point cloud registration,” in *ECCV*, 2018.
- [6] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” in *ICLRW*, 2014.
- [7] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, “Striving for simplicity: The all convolutional net,” in *ICLRW*, 2015.
- [8] T. Zheng, C. Chen, J. Yuan, and K. Ren, “Pointcloud saliency maps,” in *ICCV*, 2019.
- [9] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 Year, 1000km: The Oxford RobotCar Dataset,” *IJRR*, vol. 36, no. 1, pp. 3–15, 2017.
- [10] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *IJRR*, vol. 32, no. 11, pp. 1231 – 1237, 2013.
- [11] J. Li and G. H. Lee, “Usip: Unsupervised stable interest point detection from 3d point clouds,” in *ICCV*, 2019.
- [12] G. Georgakis, S. Karanam, Z. Wu, J. Ernst, and J. Koeck, “End-to-end learning of keypoint detector and descriptor for pose invariant 3d matching,” in *CVPR*, 2018.
- [13] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *NeurIPS*, 2015.
- [14] N. Savinov, A. Seki, L. Ladicky, T. Sattler, and M. Pollefeys, “Quad-networks: unsupervised learning to rank for interest point detection,” in *CVPR*, 2017.
- [15] T. Jakab, A. Gupta, H. Bilen, and A. Vedaldi, “Unsupervised learning of object landmarks through conditional image generation,” in *NeurIPS*, 2018.
- [16] Y. Zhang, Y. Guo, Y. Jin, Y. Luo, Z. He, and H. Lee, “Unsupervised discovery of object landmarks as structural representations,” in *CVPR*, 2018.
- [17] T. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih, “Unsupervised learning of object keypoints for perception and control,” in *NeurIPS*, 2019.
- [18] Y. Verdie, K. M. Yi, P. Fua, and V. Lepetit, “TILDE: A temporally invariant learned detector,” in *CVPR*, 2014.
- [19] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, “Lift: Learned invariant feature transform,” in *ECCV*, 2016.
- [20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation,” in *CVPR*, 2017.
- [21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space,” in *NeurIPS*, 2017.
- [22] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, “Pointcnn: Convolution on x-transformed points,” in *NeurIPS*, 2018.
- [23] M. A. Uy and G. H. Lee, “PointNetVLAD: Deep Point Cloud Based Retrieval for Large-Scale Place Recognition,” in *CVPR*, 2018.
- [24] C. Xiang, C. R. Qi, and B. Li, “Generating 3D adversarial point clouds,” in *CVPR*, June 2019.
- [25] D. Xu, D. Anguelov, and A. Jain, “PointFusion: Deep sensor fusion for 3D bounding box estimation,” in *CVPR*, 2018.
- [26] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, “Frustum PointNets for 3D object detection from RGB-D data,” in *CVPR*, 2018.
- [27] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *ICCV*, Oct 2017.
- [28] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, “Sanity checks for saliency maps,” in *NeurIPS*, 2018.
- [29] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller, “How to explain individual classification decisions,” *J. Mach. Learn. Res.*, vol. 11, pp. 1803–1831, Aug. 2010.
- [30] J. Kapur, P. Sahoo, and A. Wong, “A new method for gray-level picture thresholding using the entropy of the histogram,” *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 3, pp. 273 – 285, 1985.
- [31] Y. Zhong, “Intrinsic shape signatures: A shape descriptor for 3d object recognition,” in *ICCVW*. IEEE, 2009, pp. 689–696.

Summary

This chapter proposed a keypoint detection technique based on saliency information from a trained learned descriptor network. This can be seen as an alternative to segmentation in the framework of segment-based localization methods. The region around the keypoints was to be extracted as a possible segment to perform localization. The proposed method can be applied to any differentiable descriptor and directly maximized matchability and repeatability of keypoints in the objective function.

The crucial finding from this chapter was that keypoints extracted based on a learned descriptor parameters are already performing better than traditional keypoint detectors. This suggested that a descriptor network already possessed the required information of which regions in the scene were interesting. While the definition of a keypoint is ambiguous, this chapter provided a good intuition that keypoint detectors are specific to a descriptor. An interesting conclusion was that high-level features extracted from the first few layers of the architecture were not as useful as features extracted from later layers - four to six. Another interesting observation was that the number of RANSAC iterations could be greatly reduced by providing a number of keypoints in the same region. While this sounds intuitive, it was challenging to obtain keypoints in a similar position with other state-of-the-art learning methods.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	SKD: Keypoint Detection for Point Clouds using Saliency Estimation
Publication Status	☐ Published ☐ Accepted for Publication ☒ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Tinchev, G., Penate-Sanchez, A., & Fallon, M. (2021). SKD: Keypoint Detection for Point Clouds using Saliency Estimation. <i>IEEE Robotics and Automation Letters</i>

Student Confirmation

Student Name:	Georgi Tinchev		
Contribution to the Paper	- Primary author of the work - Developed proof-of-concept based on saliency estimation - Designed experiments and comparisons to prior art - Extended the evaluation KITTI dataset by a factor of a 100 - Evaluated the proposed approach against state-of-the-art research methods - Implemented 3D variants of 2D feature descriptors for comparison reasons - Analysed both matchability and repeatability metrics for keypoints evaluation - Investigated geometric registration of the proposed approach against baselines		
		Date	30/12/2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Dr. Maurice Fallon		
Supervisor comments: The contributions listed above are accurate		
		Date 30/12/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

7

Predicting Alignment Risk to Prevent Localization Failure

Motivated by the feature design in chapter 3, we wanted to explore whether similar geometric features can also be applied to other approaches, such as predicting when a registration is likely to fail. The following chapter presents work done in collaboration with other students. The main contribution relevant to this thesis is the *alignability* parameter defined as a geometric feature, similar to the ones considered in chapter 3. A secondary contribution is the experimental evaluation of the proposed project as well as the modular structure of the method. This chapter presents an approach capable of classifying when a localization failure may occur.

The manuscript was presented at the IEEE International Conference on Robotics and Automation, 2018 [Nobili et al. 2018].

Predicting Alignment Risk to Prevent Localization Failure

Simona Nobili^{1,2}, Georgi Tinchev² and Maurice Fallon²

Abstract—During localization and mapping the success of point cloud registration can be compromised when there is an absence of geometric features or constraints in corridors or across doorways, or when the volumes scanned only partly overlap, due to occlusions or constrictions between subsequent observations. This work proposes a strategy to predict and prevent laser-based localization failure. Our solution relies on explicit analysis of the point cloud content prior to registration. A model predicting the risk of a failed alignment is learned by analyzing the degree of spatial overlap between two input point clouds and the geometric constraints available within the region of overlap. We define a novel measure of alignability for these constraints. The method is evaluated against three real-world datasets and compared to baseline approaches. The experiments demonstrate how our approach can help improve the reliability of laser-based localization during exploration of unknown and cluttered man-made environments.

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) systems need to be able to reliably operate with a low failure rate for long periods of time and in a variety of environments. These systems should include fail-safe operation modes and auto-tuning capabilities so as to adapt to different challenges and achieve the vision of robot perception described in [1].

The work presented in this paper is focused on guaranteeing fail-safe operation of laser-based localization systems during exploration of cluttered man-made environments.

State-of-the-art laser-based localization systems have been demonstrated to achieve low drift over long distances but can also be easily induced to fail in many real-world scenarios. A cause of failure is the absence of geometric features which are necessary to constrain the alignment between two point clouds. For example, long corridors are unconstrained because of missing geometric features in one dimension. Failures can also occur when passing through doorways, or due to occlusions which cause large variations in the volume scanned by consecutive sensor sweeps. We define *alignability* as a measure of the capacity for two point clouds to be aligned given their geometric constraints, e.g., mutually visible planar surfaces.

In many cases sensors have a limited field-of-view (FOV) because they are physically integrated within a vehicle's chassis. With an obscured FOV, the degree of overlap between consecutive scans made by the sensor is gradually reduced. Occlusions and constrictions in cluttered environments also introduce a large degree of overlap variation. This

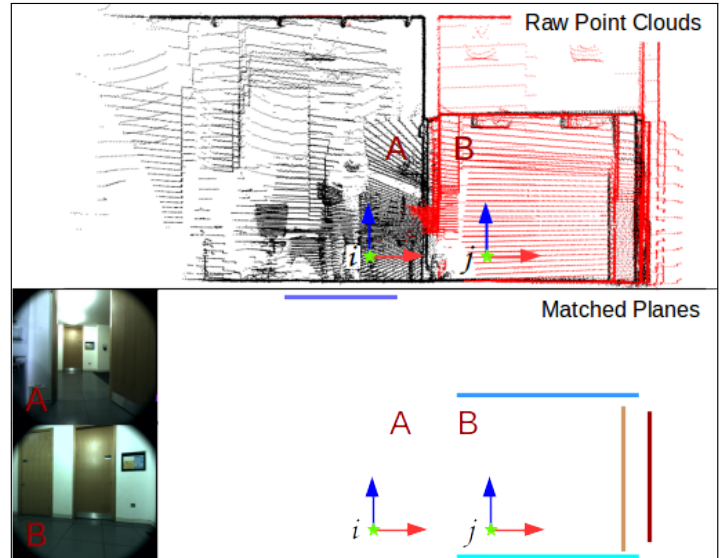


Fig. 1: The side view of a robot crossing the doorway between rooms A and B while capturing point clouds in poses i and j . According to our metric the clouds overlap by as little as 15%, and alignability is low due to the absence of matched planes in the direction of the reader. *Top*: The raw point clouds from each location. *Bottom*: Our proposed method analyses overlap and alignability, the latter derived when matching planes common to each point clouds. These are used to infer the suitability for alignment.

is a major problem for point cloud registration algorithms, as they are sensitive to the degree of overlap between input clouds. We believe that there is a need for failure prediction which is reliable to varying overlap.

The main contribution of this work is a method which predicts the *risk of failed alignment* between two point clouds, which is learned as a function of the overlap between a reference and a reading cloud¹ and the constraints available in the region of overlap between them. An example of unconstrained alignment is illustrated in Fig. 1. In contrast to previous works, [3], [4], [5], our method accounts for the degree of overlap between point clouds. It is also independent of point-wise data association and the registration approach used. The pipeline of our approach is shown in Fig. 2, which assumes that an initial (drifting) estimate of the sensor's pose is available to initialize the alignment.

Our contribution is broken down as follows:

- i) we define a novel overlap metric for 3D point clouds which takes into account the relative poses from which the clouds were captured, the structural features of the clouds, as well as the free space information. The metric gives improved performance with respect to previous work, particularly when occlusions occur,
- ii) we derive an *alignability* metric which quantifies the

¹S. Nobili is with the Institute of Perception, Action and Behavior, School of Informatics, University of Edinburgh, UK.

²All authors are with the Oxford Robotics Institute, University of Oxford, UK. {snobili, gtinchev, mfallon}@robots.ox.ac.uk

¹Using the notation from [2] we refer to the inputs to registration as a *reference* and a *reading* cloud with the latter to be aligned to the former.

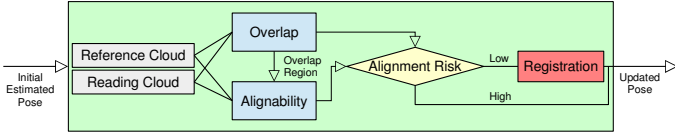


Fig. 2: The proposed pipeline.

degree to which alignment is constrained by exploiting the planar geometry commonly found in indoor environments. Our data-driven approach achieves a higher accuracy with respect to previous approaches,

- iii) we learn a model which can predict *alignment risk*, based on these overlap and alignability measures. The model allows us to prevent registration failure when the geometry is unstable and the overlap is not uniform.

The remainder of the paper is structured as follows: Sec. II presents related approaches from the literature, Sec. III describes our algorithm in detail, Sec. IV presents an extensive experimental evaluation.

II. RELATED WORK

Iterative Closest Point (ICP) is a very commonly used method for 3D point cloud alignment. Its basic formulation [6] estimates the relative alignment between two point clouds iteratively, through four main steps: data filtering, data association, outlier rejection and minimization of a point-to-point distance function. Several improvements to the algorithm have been proposed over the past years which focus on changing the minimized distance function to suit different environments [7], [8], [9].

However, the optimal registration estimated by ICP is not always a good solution. This can be because of an unconstrained degree of freedom such as when crossing a doorway (Fig. 1), or trying to align within a long corridor.

Previous works [3], [4], [5] have stated that the stability of registration can be evaluated after the set of point matches have been selected, by Principal Component Analysis (PCA) on the covariance matrix used for error minimization. If the covariance matrix is not full rank the registration is underconstrained. However, this analysis depends on the data association step of the registration algorithm. When the overlap is low, the number of point matches available for data association might be insufficient for the eigenvectors of the covariance matrix to stabilize, thus leading to an unreliable measure of the constraints. Motivated by this, in our work we focus on analysis **before** the point-wise data association step of the ICP algorithm, and we explicitly account for overlap variation.

Zhen *et al.* [10] formulated a *localizability* measure for a sensor with respect to a prior 3D map of a structure of interest. It was computed offline by generating synthetic laser data to simulate observations from within the map. They used it to plan trajectories of a UAV so as to stay within areas with high localizability. Having a different application in mind, in our work we extend their formulation to a measure of *alignability* between two subsequent clouds. This is computed online to account for the current dynamics of the scene.

Pathak *et al.* [11] recognized the sensitivity of registration to low data overlap and formulated two overlap metrics which can be used to study the reason of alignment failures. The first metric used octrees to model **occupied** regions and then derive an overlapping surface area between two point clouds. The second measured the number of pixels on range-images mutually visible from both scans. The metrics were computed given the ground-truth alignment between the clouds. Herein, we use octrees to model **occupied** and **free** space and then derive an overlapping volume between two point clouds. This makes the estimate more robust to initial misalignment.

Pomerleau *et al.* [2] formulated an overlap measure using the ratio of points from a first cloud for which there was a corresponding point in the second cloud. This approach is asymmetric, for example, when the reference cloud is larger than the reading cloud. In contrast, we define overlap so as to be resilient to this asymmetry.

Finally, in our previous work [12] we proposed a strategy for non-incremental 3D scene registration, called Auto-tuned ICP (AICP). AICP extended a baseline ICP implementation [2] to more reliably register point clouds with reduced overlap by automatically tuning an outlier-rejection filter to account for the degree of overlap of the sensor’s footprint. This framework allowed accurate registration to a single reference point cloud despite significant motion by the robot. In this work we extend AICP with a more robust overlap parameter which corrects for some issues highlighted in Sec. IV.

In summary, our work differs from the state-of-the-art in that we learn a model for predicting alignment risk which is based on both the overlap and alignability parameters. We demonstrate the utility of our model in a general context, where no prior map is available and overlap between the input point clouds varies dramatically.

In a SLAM context, we address the problem of preventing failures from within the front-end module of the system, whereas other works focus on graph optimization strategies to remove outliers at a back-end level [13], [1].

III. PREDICTING ALIGNMENT RISK

In this section, we derive a continuous variable quantifying the risk of alignment failure when registering two point clouds. We first define measures for overlap and alignability of two point clouds. We then generate a meta-parameter which can be used to predict the risk of alignment failure.

A. Measuring Overlap

We define the overlap, $\Omega \in [0, 1]$, between two point clouds using the initial estimated alignment between them (from odometry for example), their structural features and information about free space which is directly available given the sensor’s origin.

The reference cloud wC_i and the reading cloud wC_j are captured from two sensor poses i and j , expressed in the world coordinate frame w .

Using an octree structure [14], two corresponding octrees are constructed, wO_i and wO_j . Each explicitly models both free and occupied space. From wO_i and wO_j , another octree containing the set of common voxels is constructed, O^{ij} , which defines the volume of overlap between the clouds.

We define the overlap parameter as

$$\Omega = \min \left(\frac{|O^{ij}|}{|O_i|}, \frac{|O^{ij}|}{|O_j|} \right) \quad (1)$$

where $|\bullet|$ indicates the cardinality of voxels in an octree. Our volumetric representation of overlap is shown in Fig. 4 (right).

B. Measuring Alignability

We define *alignability* α as a measure of the geometric constraints which can be used to constrain alignment between a reference and a reading cloud. The alignment between a pair of 3D point clouds is well-constrained if the transform aligning them is constrained in all three dimensions. Intuitively, we envisage that at least three mutually visible non-parallel planes should exist.

In the following we describe the two steps of our strategy: (i) firstly, we match planes common to the input clouds and compute a matrix $\mathbf{N}$ as the set of normal directions extracted from these planes, (ii) secondly, we show how $\mathbf{N}$ represents the set of available constraints between the clouds. We compute α from PCA on the row vectors of $\mathbf{N}$.

Matching Plane Patches: Having segmented the reference point cloud into a set of plane patches², we select only the ones which belong to the volume of overlap between the clouds, O^{ij} . We consider the u -th patch P_u from this set. Similarly, we define P_v from the set of patches extracted from the reading cloud.

For each pair P_u and P_v , we compute a matching score Ω_p which is defined as the degree of spatial overlap between the patches. We consider each patch as a set of points contained within a bounding box in $\mathbb{R}^3$. We define the bounding boxes as V_u and V_v .

Ω_p is computed from the set of points P_u^{ij} and P_v^{ij} belonging to the patches P_u and P_v , and living in the volume of intersection between V_u and V_v , as shown in Fig. 3,

$$\Omega_p = \frac{|P_u^{ij}|}{|P_u|} \cdot \frac{|P_v^{ij}|}{|P_v|}. \quad (2)$$

In Eq. (2) $|\bullet|$ indicates the cardinality of a set.

The best match for P_v between any plane P_u is the one maximizing the overlap Ω_p . For a match to be accepted, both a condition on Ω_p and on the maximum angle between the normals must hold. The normal directions $\mathbf{n}_k^T = [n_{k_x}, n_{k_y}, n_{k_z}]$ are extracted per point $k \in [1 : M]$ (where

²We refer to plane patches as locally planar distributions of points. We adopt a region growing strategy for plane segmentation [15]. A patch is accepted only if it satisfies criteria about its planarity and dimensions (e.g., larger than 0.30×0.30 m).

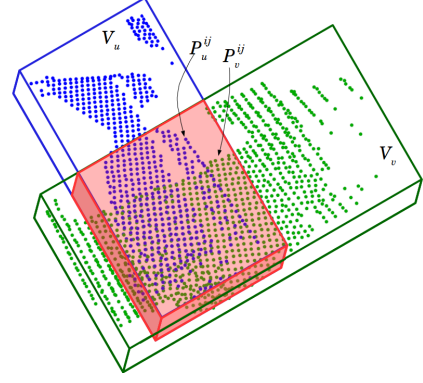


Fig. 3: Illustration of the plane matching strategy. A pair of plane patches P_u (blue points) and P_v (green points) are shown within their bounding boxes, V_u and V_v . The volume of intersection is shown in red. A score Ω_p for the match is computed from the percentage of points falling within this volume.

M is the number of points from all matched patches in the reading cloud), and $\mathbf{N} \in \mathbb{R}^{M \times 3}$ is defined as:

$$\mathbf{N} = \begin{bmatrix} n_{1_x} & n_{1_y} & n_{1_z} \\ \vdots & \vdots & \vdots \\ n_{M_x} & n_{M_y} & n_{M_z} \end{bmatrix}. \quad (3)$$

One could argue that plane matching is a variant of data association. The data association step of ICP typically involves a local point search within each pair of patches. Instead, we consider simple geometric models (the bounding boxes) to match the plane patches directly, instead of their points. This global search is easier to solve and does not depend on local density of the points. This approach will be further explored in future work by considering convex hulls as in [11].

Computing Alignability: We consider the constraints imposed on the pose of the sensor by the current measurement with respect to a previously captured point cloud. Using a formulation similar to [10], we consider two point clouds captured sequentially in time and the constraint between a current measurement point $\mathbf{p}_j$ and a measurement point $\mathbf{p}_{i,t} \in \mathbb{R}^3$ at time t in the past, both lying on the same plane:

$$\mathbf{n}_p^T (\mathbf{p}_j - \mathbf{p}_{i,t}) = 0 \quad (4)$$

where $\mathbf{n}_p$ is the plane normal. Given the robot position $\mathbf{x} \in \mathbb{R}^3$, we can formulate a second constraint:

$$\mathbf{x} + \mathbf{r}_j = \mathbf{p}_j \quad (5)$$

where $\mathbf{r}_j$ is the ray vector from the current measurement. Substituting Eq. (5) into Eq. (4) and combining the constraints imposed by all measurements in a sweep, we obtain the system of equations

$$\mathbf{N}\mathbf{x} = \mathbf{c}_j \quad (6)$$

where $\mathbf{c}_j$ is a constant vector $[c_1 \dots c_k]^T$. The matrix $\mathbf{N}$ represents the constraints which exist between the reference and reading cloud from the current sensor pose. We can identify the unconstrained dimensions of the system in Eq. (6) by

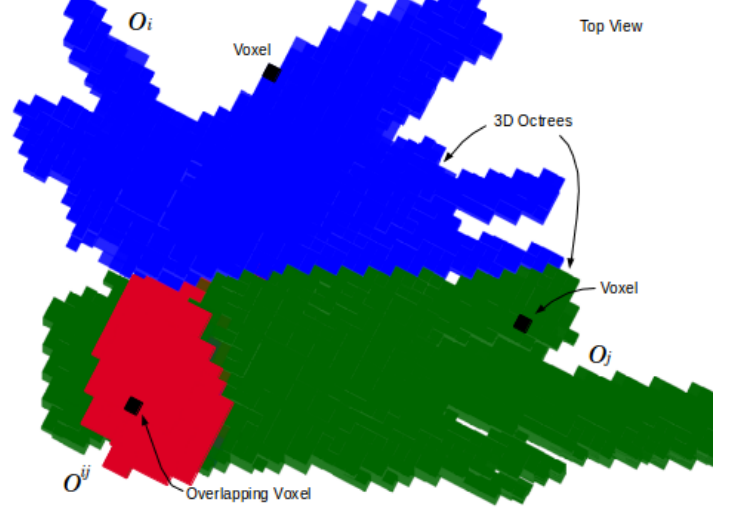
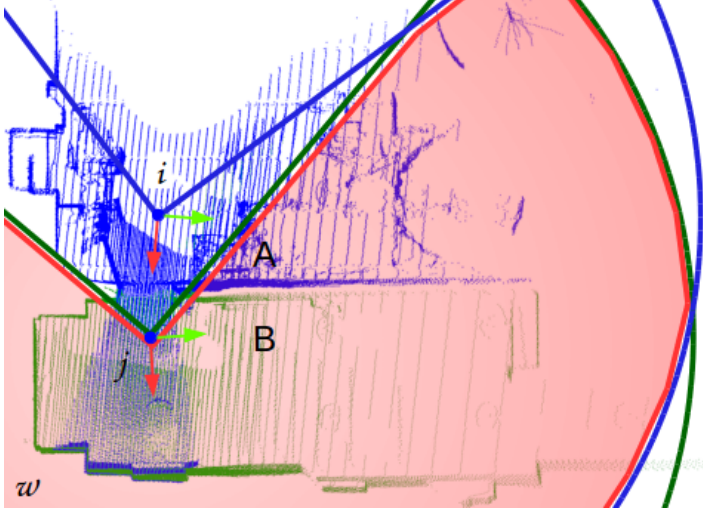


Fig. 4: *Left*: FOV-based overlap parameter [12]. A reference (blue) and reading (green) cloud have been captured from the sensor poses i and j , the latter after crossing a doorway. The region of overlap (red) is delimited by the sensor footprint. In this case the wall between room A and B occludes the view between i and j and the region of overlap is over-estimated. *Right*: The proposed octree-based overlap parameter corrects for the issues occurring in previous work, particularly when occlusions occur. Two octree structures are constructed to model free and occupied space from the two clouds. The volume shown in red, identified by our parameter, reflects what we intuitively think of as volume of overlap.

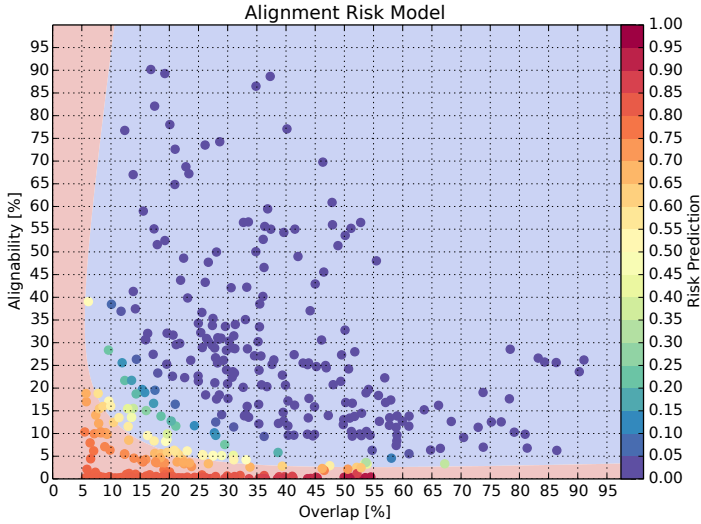


Fig. 5: Alignment risk model learned from overlap and alignability estimates. The classifier has been trained on a set of 1200 binary labeled samples (1:failure/0:success). We show the predictions on the test set, where high risk of alignment failure (red) is expected for low overlap and alignability values, following a polynomial relationship. We observe that using only one parameter (overlap or alignability) is not sufficient, e.g., just using a threshold at 5% on alignability would still accept all samples with overlap below 30%, which would risk a faulty alignment.

performing PCA on the row vectors of $\mathbf{N}$. The structure tensor is of rank 3, hence it follows that the eigenvalues, $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. In Eq. (7) we define *alignability* as the scattering of $\mathbf{N}$:

$$\alpha = \frac{\lambda_3}{\lambda_1}. \quad (7)$$

The scattering parameter is defined as the probability of a set of points to be labeled as a 3D structure [16]. As $\alpha \in [0, 1]$, low values indicate poorly constrained corresponding eigenvector in one of the dimensions.

C. Predicting Alignment Failure

Given these two individual metrics, overlap and alignability, we wish to produce a single combined value which can predict the *risk of alignment failure* $\rho \in [0, 1]$ where a score

of zero corresponds to where an alignment will be reliable while a score of one suggests an alignment which is ill-conditioned and will likely fail.

We observed that individually Ω and α cannot predict alignment failure, therefore we define ρ as a function

$$\rho = f(\Omega, \alpha). \quad (8)$$

We learned a model (Fig. 5) for ρ using a third degree polynomial Support Vector Classifier (SVC), which we found empirically to capture the function. Particularly, the prediction ρ is evaluated as the distance of a sample from the optimal hyper plane. Depending on the navigation application, the choice of ρ allows for operation at a preferred point on a Receiver Operator Characteristic (ROC) curve.

The classifier is trained on 1200 real-data observations, as described in Sec. IV. Each observation is registered using AICP and each alignment is manually labeled as a success or a failure.

IV. EXPERIMENTAL EVALUATION

So as to validate the study in this paper we carried out a series of experiments using the datasets in Tab. I, as well as a fourth dataset similar to Forum (IF) to train the classifier. The proposed metrics are compared with two baseline terms:

- *Inverse Condition Number (ICN)*: the Condition Number is used to determine whether a linear system is ill-conditioned [17]. For the linearized system $\arg \min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|$, it is computed as the ratio of the minimum and maximum eigenvalues of $\mathbf{A}^T \mathbf{A}$. For comparison purposes, we consider its inverse and denote it as *ICN*.
- *Degeneracy (D)*: for the linearized system above, the degeneracy factor measures the stiffness of the solution w.r.t. disturbances to the constraints [5]. It is computed from the eigenvalues of $\mathbf{A}^T \mathbf{A}$ as $D = \lambda_{\min} + 1$.

Our evaluation consists of four experiments:

	Stairs – ST	Apartment – AP	Forum – IF
Source	Pomerleau <i>et al.</i> [18]	Pomerleau <i>et al.</i> [18]	Ours
Situation	Indoors/Outdoors	Indoors	Indoors
Environment	Structured	Structured, variable scanned volumes	Structured, variable scanned volumes, large occlusions
Dynamics	None	Furniture displaced	Few moving people
Sensor FOV	$\sim 270^\circ \times 270^\circ$	$\sim 270^\circ \times 270^\circ$	$\sim 200^\circ \times 200^\circ$
Path	3D (up a staircase)	2D, small loops	2D, large loop
Scene Area	$(21 \times 111 \times 27)$ m	$(17 \times 10 \times 3)$ m	$\sim (32 \times 27 \times 20)$ m
Scene Sections	1 corridor, 1 staircase, 1 outdoor	5 rooms, 1 corridor	2 rooms, 2 atria, 1 corridor
# of Scans	31	45	117
# of Points	$\sim 191000/\text{scan}$	$\sim 365000/\text{scan}$	$\sim 60000/\text{scan}$
Ground Truth	✓	✓	✗
Application	Exp. C	Exp. C	Exp. D

TABLE I: Features of the datasets used for our experiments.

		True Condition	
		Alignment Fails	Alignment Succeeds
Predicted Condition	Failure	True Positive (TP)	False Positive (FP)
	Success	False Negative (FN)	True Negative (TN)

TABLE II: Confusion map used for our experiments.

Thresholds			
ICN	D	AR	
≤ 0.03	≤ 0.06	≥ 0.50	ST
≤ 0.05	≤ 0.03		AP
–	≤ 0.05		IF

TABLE III: Fine-tuned (*ICN*, *D*) and learned (*AR* – ours) thresholds for each dataset.

- A) An example comparison between the proposed point cloud overlap parameter Ω and the parameter proposed in our prior work [12]. We demonstrate that Ω is robust in the presence of occlusions.
- B) A validation of the *alignability* factor with simulated data. We show that in our experiments *alignability* can predict geometric instability more reliably than *degeneracy*.
- C) An evaluation of how the proposed measure of *alignment risk* (*AR*) outperforms *ICN* and *D* using two standardized datasets from [18]. The experiments created 2986 point cloud alignments and demonstrate the accuracy of our solution with respect to overlap variations.
- D) A demonstration of the performance of our localization system on a third dataset, where the proposed method is essential when navigating along corridors and through constrictions and environmental clutter. The system is successful when traveling along a ~ 180 m path. It prevents a total of 21 failures and allows the robot to return to the starting location with a final position error of 0.41% of the trajectory length.

In all our experiments we utilized the thresholds shown in Tab. III when predicting registration failure. We define a failure as 3D pose translational error greater than 0.02 m or rotational error greater than 1° .

A video to accompany this paper is available online.

A. Example Illustrating Overlap

To demonstrate how the proposed octree-based formulation of point cloud overlap better suits real-world scenarios we compared it to our previous FOV-based overlap parameter [12], which was based only on the shape of the sensor’s footprint. We now define overlap based on the actual structure of the data rather than the simple sensor’s FOV, as discussed in Sec. III-A. A graphical comparison between the two metrics is shown in Fig. 4.

Consider the situation depicted in Fig. 6 (top). During exploration of an indoor environment the robot moves from

one room to another through a doorway. At the start, the robot captures a point cloud (shown in blue) which will be used as the reference cloud. Thereafter the robot moves further away from the starting pose (Cloud 2) and the degree of overlap between the reference and the subsequent point clouds decreases – particularly after passing through a doorway.

The plot in Fig. 6 (bottom) shows that the FOV-based parameter over-estimates what we intuitively think of as overlap. As the robot enters the new room and turns to the left (Cloud 6), the overlap is clearly low. In contrast, the proposed octree-based parameter successfully identifies this decrease in overlap. Our subsequent experiments will solely use the octree-based overlap parameter.

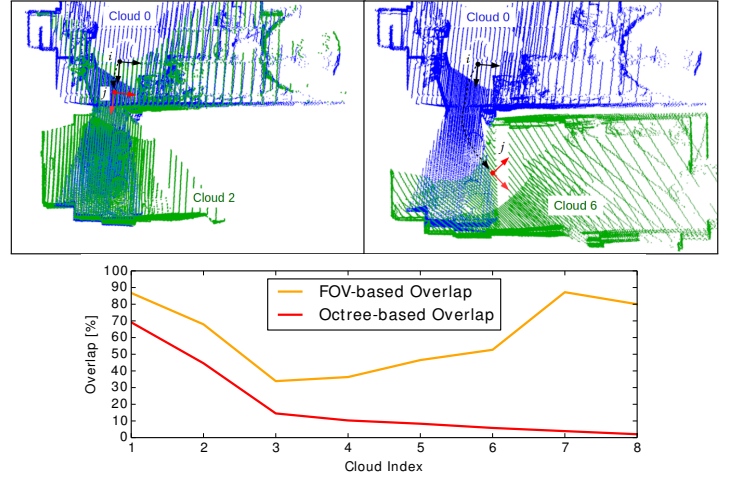


Fig. 6: Exp. A. An example where the basic FOV-based overlap is overconfident while the proposed octree-based parameter correctly estimates overlap. *Top*: Top-down view of the scenario in Fig. 1. A reference and a reading point cloud (blue and green) are captured from the first pose (black) and the current one (red), after crossing a doorway. *Bottom*: Estimate of overlap between the reference and a series of subsequent clouds. Due to its formulation, the FOV-based overlap is overconfident in such a scenario.

B. Validating the Alignability Metric

In a second experiment we aim to show that the proposed *alignability* metric can identify geometric instability in the input point clouds.

We simulate reference and reading point clouds from within a simple cube. We progressively remove surfaces from the cubic reading cloud until the alignment problem becomes unconstrained. We consider the following events, including four constrained (1-4) and five unconstrained geometries (5-9) of the reading cloud: 1) cube, 2) cube missing one surface, 3) cube missing two adjacent surfaces, 4) cube missing three adjacent surfaces and one common corner, 5) cube missing two opposite surfaces, 6) cube missing three surfaces (two opposite), 7) two opposite surfaces, 8) two adjacent surfaces, 9) one surface.

We compare the *degeneracy* and *alignability* parameters using this simulated dataset. For each event the task is to identify geometric instability between the reference and reading cloud. The task was run 100 times per event, with random initial miss-alignment sampled from a zero-mean Gaussian distribution with 0.10 m and 10° variance. The threshold for the two parameters is set to 0.06 and 10 respectively,

and values below threshold indicate unconstrained geometry. Fig. 7 illustrates the result of the experiment. The average accuracy³ of the two parameters was 79.8% for *degeneracy* and 97.7% for *alignability*. The *degeneracy* factor is computed after the point-wise data association step of ICP, which suffers from higher sensitivity to the initial perturbation. In the case of *alignability* the estimates have lower variance and the results show better separability between constrained and unconstrained events. In our approach we analyze semantic representations (the matched planes), rather than points. This makes it less sensitive to the initial alignment error and more stable overall.

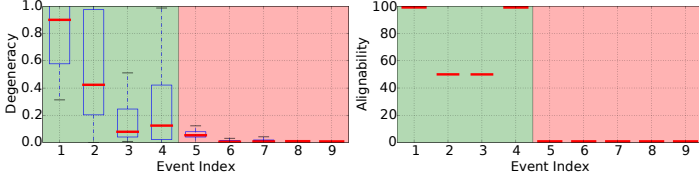


Fig. 7: Alignability validation on the simulated problem in Exp. B. The green and red backgrounds show constrained and unconstrained geometries respectively. The box plots show the quantiles 5% and 95% (bottom and top end of dashed lines), 25% and 75% (lower and higher end of blue rectangles), 50% (red bars) of the distributions. Note that the boxes for *alignability* are very narrow due to low variance.

C. Performance on Variable Overlap

In Sec. III we discussed the importance of accounting for variations in spatial overlap when predicting point cloud alignment failures in real-world scenarios. Here we evaluate the sensitivity of failure predictions to variations in the overlap between input clouds. We provide a comparison between our *alignment risk* (AR) against *ICN* and *D*.

In this experiment we use two publicly available datasets – Stairs (ST) and Apartment (AP), which consist of a Hokuyo UTM-30LX-EW planar laser mounted on a tilting unit. Ground-truth is available using an external tracking system.

For each dataset we perform a pair-wise alignment between the point clouds. We initialized the reading clouds with random perturbations sampled from a zero-mean Gaussian distribution with 0.10 m and 10° variance. We compute the overlap and measure *ICN*, *D*, *AR* for each alignment. In order to predict a registration failure, we selected a fixed threshold for *AR*, which is the optimal one learned by our model. In the case of *ICN* and *D* a threshold had to be fine-tuned specifically for each dataset, as detailed in Tab. III. The results are presented in Fig. 9.

Fig. 9a,e illustrate the matrix of overlap estimates using our proposed Ω parameter for both datasets. The diagonal elements show high overlap between each cloud with itself. Note that our estimates are symmetric by formulation.

Fig. 9b-d and Fig. 9f-h present the confusion matrices for *ICN*, *D* and *AR*, which are marked as illustrated in Tab. II.

Considering a localization task, we prefer predicting there will be a failure where there is none (false positive – **FP**) rather than the opposite (false negative – **FN**). In the regions where overlap is low *AR* predicts fewer false negatives than

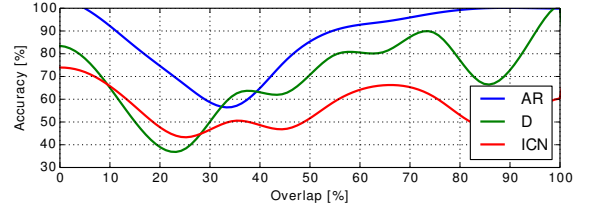


Fig. 8: Exp. C. Accuracy of failure predictions with respect to overlap variation.

ICN and *D*. For *ICN* and *D* the number of point matches is insufficient for the eigenvectors to stabilize. Our approach which measures alignability and overlap, results in a robust prediction.

ICN and *D* often predict false positives along the diagonal despite high overlap. This suggests sensitivity of the parameters to initial perturbations. On the other hand, *AR* is accurate both for low and high overlap.

Fig. 8 shows the accuracy of failure prediction as a function of overlap for each of the three parameters. The results indicate that *AR* has higher accuracy overall. As the overlap reaches 30-40% we note that the success/failure of the alignment becomes less predictable. Nevertheless, our approach performs competitively.

D. Online Performance Analysis

AICP is used as the localization framework for this last experiment, because of the increased robustness to initial alignment error and variation in overlap. The approach leverages a low drift proprioceptive-based state estimate (for example we used the wheel odometry information) to initialize the alignment. In [19], we extended the AICP algorithm to trigger a reference point cloud update when overlap decreases below an empirical threshold. In contrast, in the system presented here, we aim for continuously reliable localization by applying a registration correction to the robot state estimate depending on the predicted risk of alignment failure. When the risk is high, we query a reference point cloud update and rely on proprioception until the next laser measurement becomes available. Furthermore, we replace the original FOV-based overlap parameter with our octree-based one.

We test our localization system on the IF dataset. Our dataset is collected by a Clearpath Husky mobile robot equipped with a Carnegie Robotics Multisense SL. This sensor is composed of a stereo camera and a Hokuyo UTM-30LX-EW planar laser spinning about the forward-facing axis. Every few seconds it spins half a revolution and a 3D point cloud is accumulated. The speed of rotation of the device is set to 15RPM as a compromise between density of the clouds and accumulation time.

For this experiment we can align in three dimensions (x , y and z) and use wheel odometry to estimate roll, pitch and z . We predict registration failures using either *D* or *AR*, for comparison purposes. The robot navigates along a ~ 180 m path while exploring the area shown in Fig. 10, which includes two cluttered rooms, two wide atria with high ceilings and a corridor. The exploration involves passing across constrictions such as doorways.

³Accuracy is given by the number of true positives and true negatives by the total population.

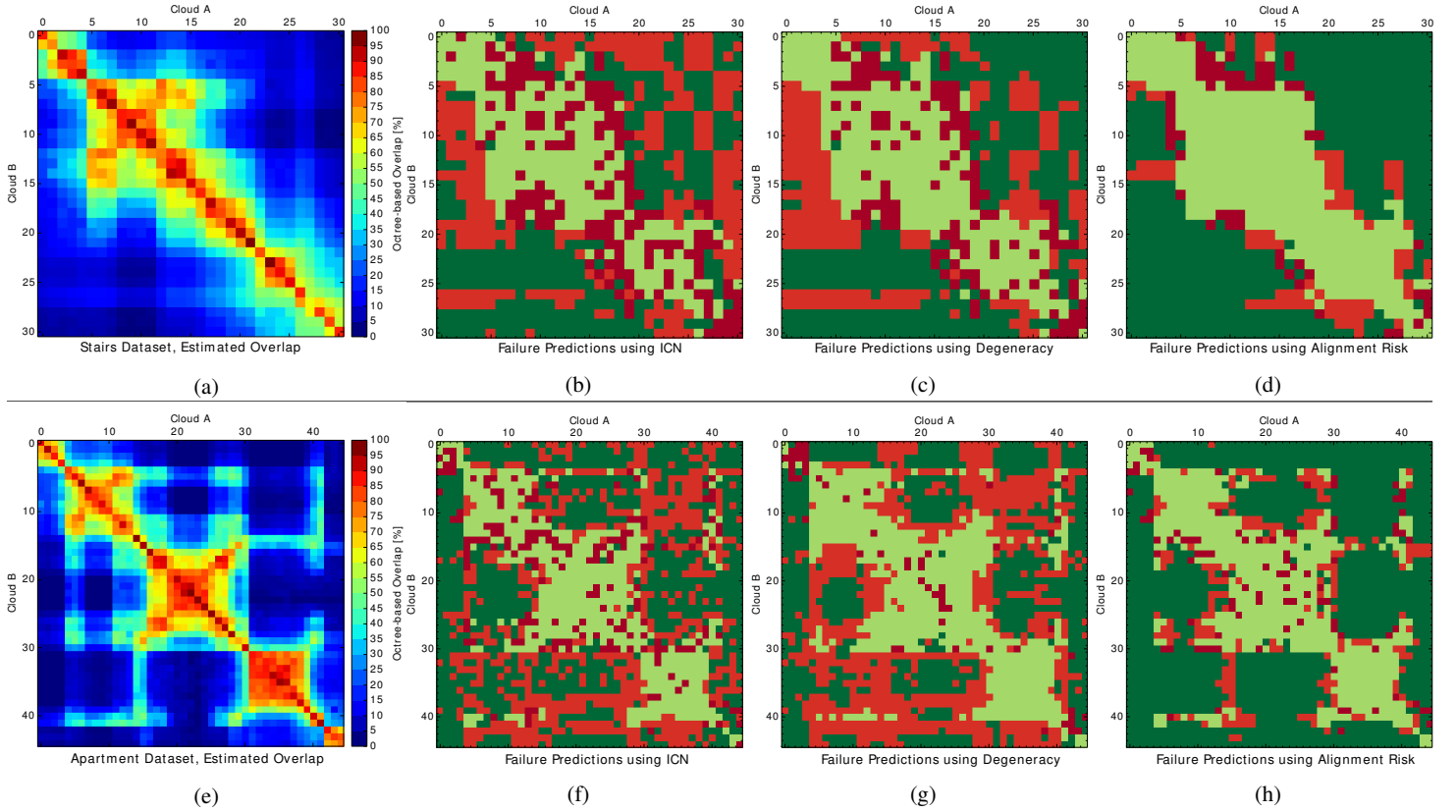


Fig. 9: Exp. C. *Left*: Estimated octree-based overlap for ST and AP datasets. Arrays can be read as the volume of common space between Cloud A and B. High overlap is marked in dark red, while low overlap in dark blue. *From right to left*: failure prediction results for AR, D, ICN on the ST (*top*) and AP (*bottom*) datasets. The matrices follow the color-coding convention in Tab. II, i.e., TP, TN, FP, FN.

The localization results, qualitatively evaluated from careful observation of the map built without loop closures in Fig. 10, are such that the estimated trajectory is close to error free when using AR. By aligning the last point cloud (red) into the first one (black) we estimate a pose error at the end of the run of 0.73 m in translation, corresponding to 0.41% of the trajectory length, and 0.71° in rotation. This result demonstrates improved localization reliability when using AR for failure prediction. The parameter could predict and prevent all failures during this experiment, while being robust to geometric instability and overlap variation occurring in a real-world scenario.

In Fig. 11 we show the registration failures identified by D (top plot) and AR (bottom plot). D could not detect the alignment failures at locations 1, 2, 3, 4 which caused the system to lose track of the robot’s trajectory. As shown in the images in Fig. 11 (right) the miss-predictions occurred in situations where the overlap between the input clouds was low, with a consequent instability of the D factor. Due to these missed detections we could not complete the run on this dataset when using D for failure prediction.

V. DISCUSSION

The proposed approach is able to reliably prevent localization failures in real-world scenarios where geometric instability and overlap variations occur frequently and challenge point cloud registration algorithms.

In our work we recognized some fundamental advantages in using a learning based approach: i) during all experiments on different datasets we selected a fixed threshold for AR.

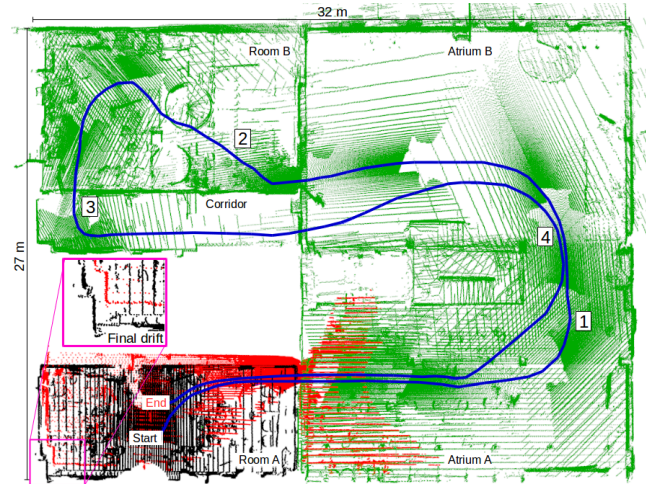


Fig. 10: Top-view of the map reconstructed during Exp. D. No loop closures have been performed. The trajectory estimated by our localization system using the proposed AR for failure prediction is depicted in blue. Along the path AR captured all 21 alignment failures. We indicate with numbers the locations where only AR was successful (constrictions – 2, 3, atria with occlusions – 1, 4), with correspondence in Fig. 11. The pose error at the end is computed from the transform which aligns the last cloud (red) into the first one (black) and results in 0.41% of the ~ 180 m trajectory length.

This threshold is the optimal one learned by our model. ii) depending on the application, we might want to be very robust to false negatives, preventing localization failures and hard recovery. The trade off for this scenario comes at the cost of accepting more false positives and reference cloud updates. In turn, each reference cloud update introduces a small incremental error to our estimate. This is useful when a low drift odometry prior is available, allowing us to

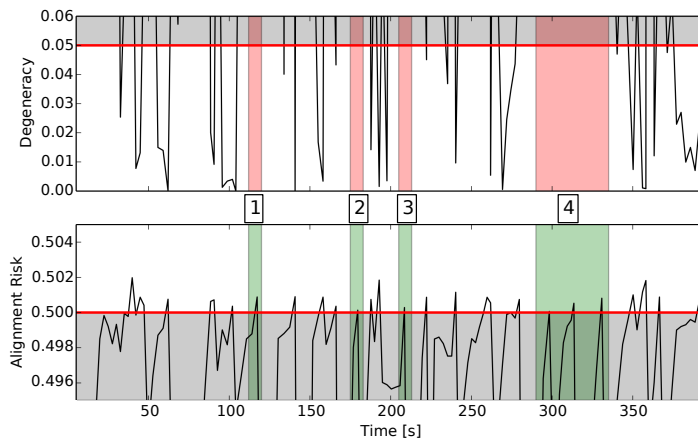


Fig. 11: *Left*: Failure predictions during Exp. D. During navigation D could not detect 4 alignment failures, as highlighted by the red sections on the top plot. We indicate each event with a number (1, 2, 3, 4). A top view of these events is shown on the right. The trajectory estimated using D for failure prediction is depicted in green, whereas the one estimated using the proposed AR is blue. AR successfully predicted all 21 failures.

lower the AR threshold to accommodate more false positive predictions.

Additionally, we believe that our model could facilitate the search of loop closures in a SLAM system, by sorting the candidates by AR score prior to data association.

VI. CONCLUSIONS

In this work we proposed a strategy for point cloud alignment failure prediction. We explored the degree to which alignment failure is affected by geometric instability of the input point clouds, as well as the spatial overlap between them. We adopted a data-driven approach to evaluate the geometric constraints available for alignment and the volume of spatial overlap between the clouds. We used this data to learn a model to predict the risk of a failed alignment.

This allows us to be independent of the adopted registration strategy and point-wise data association, as well as to easily select the optimal threshold learned by our model in order to predict a failure, which avoids manual fine-tuning.

We evaluated our approach on different datasets and provided comparisons to existing techniques. Our algorithm overcomes the weaknesses of the baseline techniques identified in the context of real-world scenarios, where constrictions and occlusions cause reduced overlap between observations. We demonstrated how our approach can help improve the reliability of laser-based localization systems during exploration of unknown and cluttered man-made environments. In a large indoor exploration demo the system was able to reliably estimate the robot state with a final pose error of 0.41% of the trajectory length, and to build an accurate 3D representation of the environment.

Future work will focus on the extension of this approach to a SLAM system with loop closure detection.

REFERENCES

- [1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.
- [2] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, Apr. 2013.
- [3] J. Guehring, "Reliable 3D surface acquisition, registration and validation using statistical error models," in *Intl. Conf. on 3-D Digital Imaging and Modeling (3DIM)*, May 2001, pp. 224–231.
- [4] N. Gelfand, L. Ikemoto, S. Rusinkiewicz, and M. Levoy, "Geometrically stable sampling for the ICP algorithm," in *Intl. Conf. on 3-D Digital Imaging and Modeling (3DIM)*, Oct 2003, pp. 260–267.
- [5] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2016, pp. 809–816.
- [6] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.
- [7] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vision Computing*, vol. 10, no. 3, pp. 145–155, Apr. 1992.
- [8] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems (RSS)*, June 2009.
- [9] J. Serafin and G. Grisetti, "NIPC: Dense normal based point cloud registration," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 742–749.
- [10] W. Zhen, S. Zeng, and S. Soberer, "Robust localization and localizability estimation with a rotating laser scanner," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2017, pp. 6240–6245.
- [11] K. Pathak, D. Borrmann, J. Elseberg, N. Vaskevicius, A. Birk, and A. Nüchter, "Evaluation of the robustness of planar-patches based 3D-registration using marker-based ground-truth in an outdoor urban scenario," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2010, pp. 5725–5730.
- [12] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based ICP tuning for robust localization of a humanoid robot," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, May 2017, pp. 4721–4728.
- [13] M. Pfingsthorn and A. Birk, "Generalized graph SLAM," *Intl. J. of Robotics Research*, vol. 35, no. 6, pp. 601–630, May 2016.
- [14] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, Apr. 2013.
- [15] T. Rabbani, F. van den Heuvel, and G. Vosselman, "Segmentation of point clouds using smoothness constraints," *ISPRS Annals of the Phot., Remote Sens. and Spatial Inf. Sciences*, vol. 35, pp. 248–253, 2006.
- [16] M. Weinmann, B. Jutzi, and C. Mallet, "Semantic 3D scene interpretation: A framework combining optimal neighborhood size selection with relevant features," *ISPRS Annals of the Phot., Remote Sens. and Spatial Inf. Sciences*, vol. 2, no. 3, pp. 181–188, August 2014.
- [17] E. W. Cheney and D. R. Kincaid, *Numerical Mathematics and Computing*, 6th ed., 2007.
- [18] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, "Challenging data sets for point cloud registration algorithms," *Intl. J. of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, Dec. 2012.
- [19] S. Nobili, M. Camurri, V. Barasuol, M. Focchi, D. Caldwell, C. Semini, and M. Fallon, "Heterogeneous sensor fusion for accurate state estimation of dynamic legged robots," in *Robotics: Science and Systems (RSS)*, July 2017.

Summary

This section focused on predicting when an alignment failure might occur when registering two LIDAR scans. The chapter detailed scenarios where such a method will be useful by complementing a full SLAM system.

The key observation was that overlap alone is not a sufficient criteria to predict when two consecutive scans will produce an ill-defined registration. Furthermore, the geometry of the scene carried information when a registration was likely to fail. In combination, a classifier considering both metrics produced a reliable way of predicting these failure cases. It is interesting to observe that the features considered for localization in both urban and natural environments are useful to also determine failure registration cases.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Predicting Alignment Risk to Prevent Localization Failure
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Nobili, S., Tinchev, G., & Fallon, M. (2018, May). Predicting alignment risk to prevent localization failure. In 2018 <i>IEEE International Conference on Robotics and Automation (ICRA)</i> (pp. 1003-1010). IEEE.

Student Confirmation

Student Name:	Georgi Tinchev		
Contribution to the Paper	- Collaborated with the main author designing the <i>alignability</i> parameter - Helped converting the code repository into separate modules - Discussed and designed different classifier options - Wrote and corrected parts of the manuscript		
	Date	30/12/2020	

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Dr. Maurice Fallon		
Supervisor comments: The contributions listed above are accurate		
	Date	30/12/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

8

Online LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure

Motivated by the success of the proposed localization system in chapter 4, recent advancements in LIDAR odometry by Nobili et al. [2017], and the proposed verification system of chapter 7, the following chapter combines the approaches into a single SLAM system. The work done was a part of a collaboration between a post-doctorate fellow and a fourth year student. The main contribution, relevant this thesis is two-fold: i) transforming a proof-of-concept deep learning model for feature learning to a loop closure algorithm chapter 4 and ii) aiding the main author in bringing down the complexity of the verification algorithm from chapter 7. The following chapter presents an application of the main approach in this thesis, ESM, as a deeply learned loop closure detection algorithm incorporated as part of a pose graph SLAM system that is deployed on a quadrupedal robot in an industrial setting.

This work was presented at the IEEE International Conference on Robotics and Automation, 2020 [Ramezani et al. 2020].

Online 3D LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure

Milad Ramezani, Georgi Tinchev, Egor Iuganov and Maurice Fallon

Abstract—In this paper, we present a factor-graph 3D LiDAR-SLAM system which incorporates a state-of-the-art deeply learned feature-based loop closure detector to enable a legged robot to localize and map in industrial environments. Point clouds are accumulated using an inertial-kinematic state estimator before being aligned using ICP registration. To close loops we use a loop proposal mechanism which matches individual *segments* between clouds. We trained a descriptor offline to match these segments. The efficiency of our method comes from carefully designing the network architecture to minimize the number of parameters such that this deep learning method can be deployed in real-time using only the CPU of a legged robot, a major contribution of this work. The set of odometry and loop closure factors are updated using pose graph optimization. Finally we present an efficient risk alignment prediction method which verifies the reliability of the registrations. Experimental results at an industrial facility demonstrated the robustness and flexibility of our system, including autonomous following paths derived from the SLAM map.

I. INTRODUCTION

Robotic mapping and localization have been heavily studied over the last two decades and provide the perceptual basis for many different tasks such as motion planning, control and manipulation. A vast body of research has been carried out to allow a robot to determine where it is located in an unknown environment, to navigate and to accomplish tasks robustly online. Despite substantial progress, enabling an autonomous mobile robot to operate robustly for long time periods in complex environments, is still an active research area.

Visual SLAM has shown substantial progress [1], [2], with much work focusing on overcoming the challenge of changing lighting variations [3]. Instead, in this work we focus on LiDAR as our primary sensor modality. Laser measurements are actively illuminated and precisely sense the environment at long ranges which is attractive for accurate motion estimation and mapping. Hence, we focus on the LiDAR-SLAM specifically for legged robots.

The core odometry of our SLAM system is based on Iterative Closest Point (ICP), a very well-known method for 3D shape registration [4], to register clouds from a LiDAR (Velodyne) sensor. Our approach builds upon our previous work of Auto-tuning ICP (AICP), originally proposed in [5], which analyzes the content of incoming clouds to robustify registration. Initialization of AICP is provided by the robot's state estimator [6], which estimates the robot pose by fusing the inertial measurements with the robot's kinematics

The authors are with the Oxford Robotics Institute (ORI), University of Oxford, UK. {milad, gtinchev, mfallon}@robots.ox.ac.uk, egor.iuganov@wadham.ox.ac.uk

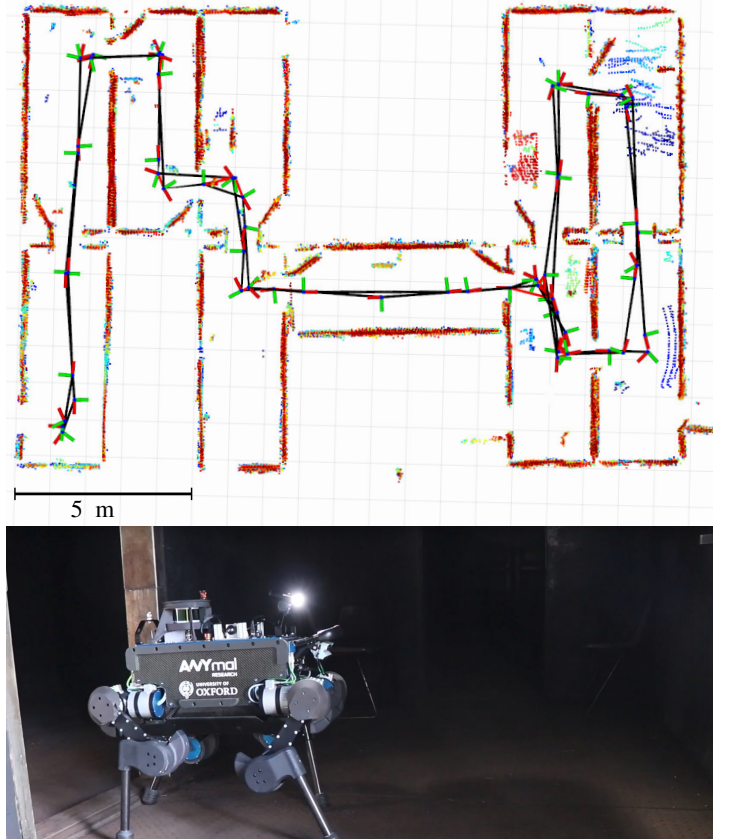


Fig. 1: Bird's-eye view of a map constructed by the ANYmal exploring an unlit, windowless, industrial facility with our proposed 3D SLAM system. The approach can rapidly detect and verify loop-closures (red lines) so as to construct an accurate map.

and joint information in a recursive approach. Our LiDAR odometry has drift below 1.5% of distance traveled.

The contributions of our paper are summarized as follows:

- 1) An online 3D LiDAR-SLAM system for legged robots based on ICP registration.
- 2) A verification metric which quantifies the reliability of point cloud registration.
- 3) A demonstration of learned loop-closure detection designed to be deployable on a mobile CPU during run time.
- 4) A real-time demonstration of the algorithm on the ANYmal quadruped robot.

The remainder of this paper is structured as follows: Sec. II presents related works. Sec. III details different components of our SLAM system. Sec. IV presents evaluation studies before a conclusion and future works are drawn in Sec. V.

II. RELATED WORKS

This section provides a literature review of perception in walking robots and LiDAR-SLAM systems in general.

A. Perception systems on walking robots

Simultaneous Localization and Mapping (SLAM) is a key capability for walking robots and consequently their autonomy. Example systems include, the mono visual SLAM system of [7] which ran on the HRP-2 humanoid robot and is based on an Extended Kalman Filter (EKF) framework. In contrast, [8] leveraged a particle filter to estimate the posterior of their SLAM method, again on HRP-2. Oriolo *et al.* [9] demonstrated visual odometry on the Nano bipedal robot by tightly coupling visual information with the robot's kinematics and inertial sensing within an EKF. However, these approaches acquired only a sparse map of the environment on the fly, substantially limiting the robot's perception.

Works such as [10] and [11] leveraged the frame-to-model visual tracking of KinectFusion [12] or ElasticFusion [13], which use a coarse-to-fine ICP algorithm. Nevertheless, vision-based SLAM techniques struggle in varying illumination conditions. While this has been explored in works such as [14] and [3], it remains an open area of research. In addition, the relatively short range of visual observations limit their performance for large-scale operations.

Compared to bipedal robots, quadrupedal robots have better versatility and resilience when navigating challenging terrain. Thus, they are more suited to long-term tasks such as inspection of industrial sites. In the extreme case, a robot might need to jump over a terrain hurdle. Park *et al.*, in [15], used a Hokuyo laser range finder on the MIT Cheetah-2 to enable the robot to jump over hurdles as high as 40 cm, though the LiDAR measurements were not used for localization.

Nobili *et al.* [16] presented a state estimator for the Hydraulic Quadruped robot (HyQ). Based on a modular inertial-driven EKF, the robot's base link velocity and position were propagated using measurements from modules including leg odometry, visual odometry and LiDAR odometry. Nevertheless, the system is prone to drift, as the navigation system did not include a mechanism to detect loop-closures.

B. LiDAR-SLAM Systems

An effective LiDAR-based approach is LiDAR Odometry and Mapping (LOAM) [17]. LOAM extracts edge and surface point features in a LiDAR cloud by evaluating the roughness of the local surface. The features are re-projected to the start of the next scan based on a motion model, with point correspondences found within this next scan. Finally, the 3D motion is estimated recursively by minimizing the overall distances between point correspondences. LOAM achieves high accuracy and low cost of computation.

Similar to LOAM, Deschaud in [18] developed a LiDAR odometry, IMLS-LiDAR, with better accuracy by leveraging scan-to-model matching.

Shan *et al.* in [19] extended LOAM with Lightweight and Ground-Optimized LiDAR Odometry and Mapping (LeGO-LOAM) which added latitudinal and longitudinal parameters separately in a two-phase optimization. In addition, LeGO-LOAM detected loop-closures using an ICP algorithm to create a globally consistent pose graph using iSAM2 [20].

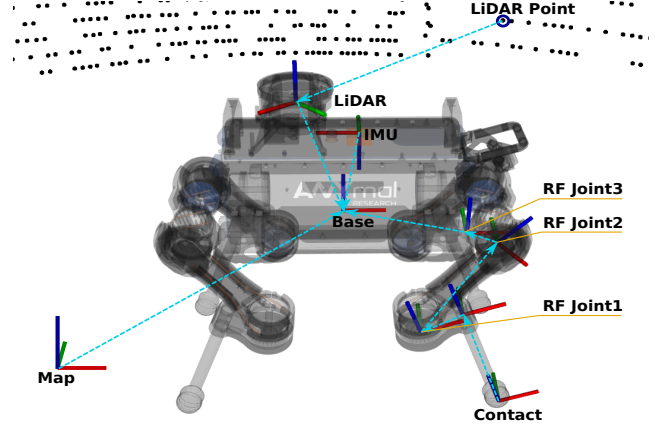


Fig. 2: Axis conventions of various frames used in the LiDAR-SLAM system and their relationship with respect to the base frame. As an example, we show only the Right Front (RF) leg.

Behley *et al.* in [21] proposed a surfel-based LiDAR-SLAM system by leveraging the projective data association between the current scan and a rendered model view from the surfel map.

Dubé *et al.* in [22] developed an online cooperative LiDAR-SLAM system. Using two and three robots, each equipped with 3D LiDAR, an incremental sparse pose-graph is populated by successive place recognition constraints. The place recognition constraints are identified utilizing the SegMatch algorithm [23], which represents LiDAR clouds as a set of compact yet discriminative features. The descriptors were used in a matching process to determine loop closures.

III. APPROACH

Our goal is to provide the ANYMAL with a drift-free localization estimate over a very long mission, as well as to enable the robot to accurately map its surroundings. Fig. 2 and Fig. 3 elucidate the different components of our system from a hardware and a software perspective.

A. Kinematic-Inertial Odometry

The proprioceptive sensors are tightly coupled using a recursive error-state estimator, Two-State Implicit Filter (TSIF) [24], which estimates the incremental motion of the robot. The position of the contact feet in the inertial frame $\{\mathcal{I}\}$, obtained from forward kinematics, is treated as a temporal measurement to estimate the robot's pose in the fixed odometry frame, $\{\mathcal{O}\}$:

$$\mathbf{T}_B^{\mathcal{O}} = \begin{bmatrix} \mathbf{R}_B^{\mathcal{O}} & \mathbf{p}_B^{\mathcal{O}} \\ 0 & 1 \end{bmatrix}, \quad (1)$$

where $\mathbf{T}_B^{\mathcal{O}} \in SE(3)$ is the transformation from the base frame $\{\mathcal{B}\}$ to the odometry frame $\{\mathcal{O}\}$.

This estimate of the base frame drifts over time as it does not use any exteroceptive sensors and the position of the quadruped and its rotation around z -axis are not observable. In the following, we define our LiDAR-SLAM system for estimation of the robot's pose with respect to the map frame $\{\mathcal{M}\}$ which is our goal. It is worth noting that the TSIF framework estimates the covariance of the state [24] which we employ during our geometric loop-closure detection.

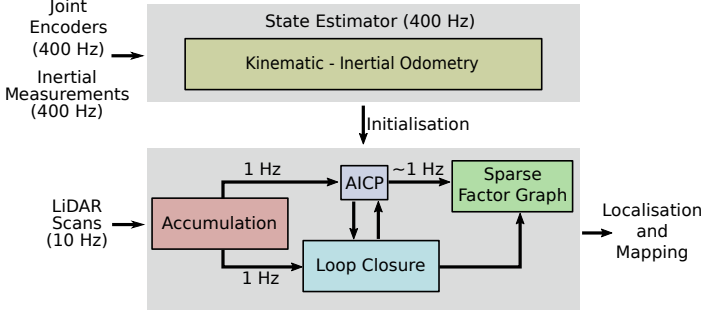


Fig. 3: Block diagram of the LiDAR-SLAM system.

B. LiDAR-SLAM System

Our LiDAR-SLAM system is a pose graph SLAM system built upon our ICP registration approach called Autotuned-ICP (AICP) [5]. AICP automatically adjusts the outlier filter of ICP by computing an overlap parameter, $\Omega \in [0, 1]$ since the assumption of a constant overlap, which is conventional in the standard outlier filters, violates the registration in real scenarios.

Given the initial estimated pose from the kinematic-inertial odometry, we obtain a **reference** cloud to which we align each consecutive **reading**¹ point cloud.

In this manner the successive reading clouds are precisely aligned with the reference clouds with greatly reduced drift. The robot's pose, corresponding to each reading cloud is obtained as follows:

$$\mathbf{T}_B^{\mathcal{M}} = \Delta_{aicp} \mathbf{T}_B^{\mathcal{O}}, \quad (2)$$

where $\mathbf{T}_B^{\mathcal{M}}$ is the robot's pose in the map frame $\{\mathcal{M}\}$ and Δ_{aicp} is the alignment transformation calculated by AICP.

Calculating the corrected poses, corresponding to the point clouds, we compute the relative transformation between the successive reference clouds to create the odometry factors of the factor graph which we introduce in Eq. (3).

The odometry factor, $\phi_i(\mathcal{X}_{i-1}, \mathcal{X}_i)$, is defined as:

$$\phi_i(\mathcal{X}_{i-1}, \mathcal{X}_i) = (\mathbf{T}_{B_{i-1}}^{\mathcal{M}^{-1}} \mathbf{T}_{B_i}^{\mathcal{M}})^{-1} \tilde{\mathbf{T}}_{B_{i-1}}^{\mathcal{M}^{-1}} \tilde{\mathbf{T}}_{B_i}^{\mathcal{M}}, \quad (3)$$

where $\tilde{\mathbf{T}}_{B_{i-1}}^{\mathcal{M}}$ and $\tilde{\mathbf{T}}_{B_i}^{\mathcal{M}}$ are the AICP estimated poses of the robot for the node $\mathcal{X}_{i-1}$ and $\mathcal{X}_i$, respectively, and $\mathbf{T}_{B_{i-1}}^{\mathcal{M}}$ and $\mathbf{T}_{B_i}^{\mathcal{M}}$ are the noise-free transformations.

A prior factor, $\phi_0(\mathcal{X}_0)$, which is taken from the pose estimate of the kinematic-inertial odometry, is initially added to the factor graph to set an origin and a heading for the robot within the map frame $\{\mathcal{M}\}$.

To correct for odometry drift, loop-closure factors are added to the factor graph once the robot revisits an area of the environment. We implemented two approaches for proposing loop-closures: a) geometric proposal based on the distance between the current pose and poses already in the factor graph which is useful for smaller environments, and b) a learning approach for global loop-closure proposal, detailed in Sec III-C.1, which scales to large environments. Each proposal provides an initial guess, which is refined with ICP.

¹Borrowing the notation from [25], the **reference** and **reading** clouds correspond to the robot's poses at the start and end of each edge in the factor graph and AICP registers the latter to the former.

Each individual loop closure becomes a factor and is added to the factor graph. The loop-closure factor, in this work, is a factor whose end is the current pose of the robot and whose start is one of the reference clouds, stored in the history of the robot's excursion. The nominated reference cloud must meet two criteria of nearest neighborhood and sufficient overlap with the current reference cloud. An accepted loop-closure factor, $\phi_j(\mathcal{X}_M, \mathcal{X}_N)$, is defined as:

$$\phi_j(\mathcal{X}_M, \mathcal{X}_N) = (\mathbf{T}_{B_M}^{\mathcal{M}^{-1}} \mathbf{T}_{B_N}^{\mathcal{M}})^{-1} (\Delta_{j,aicp} \tilde{\mathbf{T}}_{B_M}^{\mathcal{M}})^{-1} \tilde{\mathbf{T}}_{B_N}^{\mathcal{M}}, \quad (4)$$

where $\tilde{\mathbf{T}}_{B_M}^{\mathcal{M}}$ and $\tilde{\mathbf{T}}_{B_N}^{\mathcal{M}}$ are the robot's poses in the map frame $\{\mathcal{M}\}$ corrected by AICP with respect to the reference cloud $(M-1)$ and the reference cloud $(N-1)$, respectively. The $\Delta_{j,aicp}$ is the AICP correction between the current reference cloud M and the nominated reference cloud N .

Once all the factors, including odometry and loop-closure factors, have been added to the factor graph, we optimize the graph so that we find the Maximum A Posteriori (MAP) estimate for the robot poses corresponding to the reference clouds. To carry out this inference over the variables $\mathcal{X}_i$, where i is the number of the robot's pose in the factor graph, the product of all the factors, must be maximized:

$$\mathcal{X}^{MAP} = \underset{\mathcal{X}}{\operatorname{argmax}} \prod_i \phi_i(\mathcal{X}_{i-1}, \mathcal{X}_i) \prod_j \phi_j(\mathcal{X}_M, \mathcal{X}_N). \quad (5)$$

Assuming that factors follow a Gaussian distribution and all measurements are only corrupted with white noise, i.e. noise with normal distribution and zero mean, the optimization problem in Eq. (5) is equivalent to minimizing a sum of nonlinear least squares:

$$\mathcal{X}^{MAP} = \underset{\mathcal{X}}{\operatorname{argmin}} \sum_i \|y_i(\mathcal{X}_i, \mathcal{X}_{i-1}) - m_i\|_{\Sigma_i}^2 + \sum_j \|y_j(\mathcal{X}_M, \mathcal{X}_N) - m_j\|_{\Sigma_j}^2, \quad (6)$$

where m , y and Σ denote the measurements, their mathematical model and the covariance matrices, respectively.

As noted, the MAP estimate is only reliable when the residuals in Eq. (6) follow the normal distribution. However, ICP is susceptible to failure in the absence of geometric features, e.g. in corridors or door entries, which can have a detrimental effect when optimizing the pose graph. In Sec. III-D, we propose a fast verification technique for point cloud registration to detect possible failure of the AICP registration.

C. Loop Proposal Methods

This section introduces our learned loop-closure proposal and geometric loop-closure detection.

1) *Deeply-learned Loop Closure Proposal*: We use the method of Tinchev *et al.* [26], which is based on matching individual segments in pairs of point clouds using a deeply-learned feature descriptor. Its specific design uses a shallow network such that it does not require a GPU during run-time inference on the robot. We present a summary of the method called Efficient Segment Matching (ESM), but refer the reader to [26].

Algorithm 1: Improved Risk Alignment Prediction.

```

1 input: point clouds  $\mathcal{C}_S, \mathcal{C}_T$ ; estimated poses  $\mathcal{X}_S, \mathcal{X}_T$ 
2 output: alignment risk  $\rho = f(\Omega, \alpha)$ 
3 begin
4   Segment  $\mathcal{C}_S$  and  $\mathcal{C}_T$  into a set of planes:  $P_{Si}$  and  $P_{Tj}$ ,
5   Compute centroid of each plane:  $\mathbf{K}_{Si}, \mathbf{K}_{Tj}$ ,
6   Transform query keypoints  $\mathbf{K}_{Tj}$  into the space of  $\mathcal{C}_S$ ,
7   Search the nearest neighbour of each query plane  $P_{Tj}$ ,
8   for plane  $P_T$  do
9     Find match  $P_S$  amongst candidates in the k-d tree,
10    Compute the matching score  $\Omega_p$ ,
11    if  $\Omega_p$  is max then
12      Determine the normal of plane  $P_T$ ,
13      Push back the normal into the matrix  $\mathbf{N}$ ,
14    end
15  end
16  Compute alignability  $\alpha = \lambda_{max} / \lambda_{min}$ ;
17  Learn  $\rho = f(\Omega, \alpha)$ ;
18  Return  $\rho$ ;
19 end

```

First, a neural network is trained offline using individual LiDAR observations (segments). By leveraging odometry in the process, we can match segment instances without manual intervention. The input to the network is a batch of triplets - anchor, positive and negative segments. The anchor and positive samples are the same object from two successive Velodyne scans, while the negative segment is a segment chosen ≈ 20 m apart. The method then performs a series of X-conv operators directly on raw point cloud data, based on PointCNN [27], followed by three fully connected layers, where the last layer is used as the descriptor for the segments.

During our trials, when the SLAM system receives a new reference cloud, it is pre-processed and then segmented into a collection of point cloud clusters. For each segment in the reference cloud, a descriptor vector is computed with an efficient TensorFlow C++ implementation by performing a forward pass using the weights from the already trained model. This allows a batch of segments to be pre-processed simultaneously with zero-meaning and normalized variance and then forward passed through the trained model. We use a three dimensional tensor as input to the network - the length is the number of segments in the current point cloud, the width represents a fixed-length down-sampled vector of all the points in an individual segment, and the height contains the x, y and z values. Due to the efficiency of the method, we need not split the tensor into mini-batches, allowing us to process the full reference cloud in a single forward pass.

Once the descriptors for the reference cloud are computed, they are compared to the map of previous reference clouds. ESM uses an l_2 distance in feature space to detect matching segments and a robust estimator to retrieve a 6DoF pose. This produces a transformation of the current reference cloud with respect to the previous reference clouds. The transformation is then used in AICP to add a loop-closure as a constraint to the graph-based optimization. Finally, ESM's map representation is updated, when the optimization concludes.

2) *Geometric Loop-Closure Detection:* To geometrically detect loop-closures, we use the covariance of the legged

state estimator (TSIF) to define a dynamic search window around the current pose of the robot. Then the previous robot's poses, which reside within the search window, are examined based on two criteria: nearest neighborhood and verification of cloud registration (described in Sec. III-D). Finally, the geometric loop closure is computed between the current cloud and the cloud corresponding to the nominated pose using AICP.

D. Fast Verification of Point Cloud Registration

This section details a verification approach for ICP point cloud registration to determine if two point clouds can be safely registered. We improve upon our previously proposed alignability metric in [28] with a much faster method.

Method from [28]: First, the point clouds are segmented into a set of planar surfaces. Second, a matrix $\mathbf{N} \in \mathbb{R}^{M \times 3}$ is computed, where each row corresponds to the normal of the planes ordered by overlap. M is the number of matching planes in the overlap region between the two clouds. Finally, the alignability metric, α is defined as the ratio between the smallest and largest eigenvalues of $\mathbf{N}$.

The matching score, Ω_p , is computed as the overlap between two planes, P_{Tj} and P_{Si} where $i \in N_S$ and $j \in N_T$. N_S and N_T are the number of planes in the input clouds. In addition, in order to find the highest possible overlap, the algorithm iterates over all possible planes from two point clouds. This results in overall complexity $O(N_S N_T (N_{P_S} N_{P_T}))$, where N_{P_S} and N_{P_T} are the average number of points in planes of the two clouds.

Proposed Improvement: To reduce the point-wise computation, we first compute the centroids of each plane $\mathbf{K}_{Si}$ and $\mathbf{K}_{Tj}$ and align them from point cloud $\mathcal{C}_T$ to $\mathcal{C}_S$ given the computed transformation. We then store the centroids in a k-d tree and for each query plane $P_T \in \mathcal{C}_T$ we find the K nearest neighbors. We compute the overlap for the K nearest neighbors, and use the one with the highest overlap. This results in $O(N_T K (N_{P_S} N_{P_T}))$, where $K \ll N_s$. In practice, we found that $K = 1$ is sufficient for our experiments. Furthermore, we only store the centroids in a k-d tree, reducing the space complexity.

We discuss the performance of this algorithm, as well as its computational complexity in the experiment section of this paper. Pseudo code of the algorithm is available in Algorithm 1.

IV. EXPERIMENTAL EVALUATION

We employ a state-of-the-art quadruped, ANYmal (version B) [29], as our experimental platform. Fig. 4 shows a view of the ANYmal robot. The robot weighs about 33 kg without

Sensor	Model	Frequency	Specifications
IMU	Xsens MTi-100	400	Bias Repeatability: $< 0.5^\circ/s$; 5 mg
			Bias Stability: $10^\circ/h$; $40\text{ }\mu\text{g}$
LiDAR Unit	Velodyne VLP-16	10	Resolution in Azimuth: $< 0.4^\circ$ Resolution in Zenith: 2.0° Range $< 100\text{ m}$ Accuracy: $\pm 3\text{ cm}$
Encoder	ANYdrive	400	Resolution $< 0.025^\circ$
Torque	ANYdrive	400	Resolution $< 0.1\text{ Nm}$

TABLE I: Specifications of the sensors installed on the ANYmal.

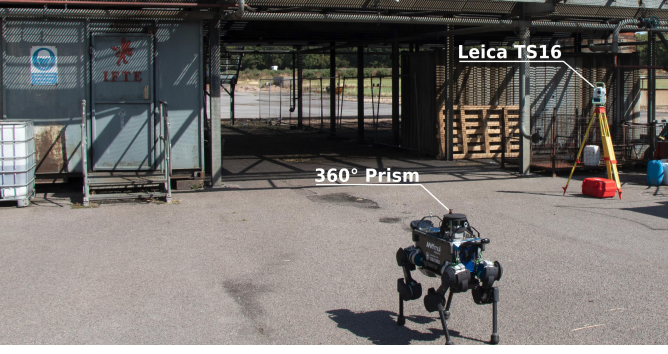


Fig. 4: Our experiments were carried out with the ANYmal, in the Oil Rig training site. To determine ground truth robot poses, we used a Leica TS16 laser tracker to track a 360° prism on the robot.

any external perception modules and can carry a maximum payload of 10 kg at the maximum speed of 1.0 m/s. As shown in Fig. 2, each leg contains 3 actuated joints which altogether gives 18 degrees of freedom, 12 actuated joints and the 6 DoF robot base, which the robot uses to dynamically navigate challenging terrain. Tab. I summarizes the specifications of the robot’s sensors used in this paper. The proposed LiDAR-SLAM system is evaluated using the datasets collected by our ANYmal quadruped robot.

We first analyze the verification method. Second, we investigate the learned loop-closure detection in terms of speed and reliability (Sec. IV-B). We then demonstrate the performance of our SLAM system on two large-scale experiments, one indoor and one outdoor (Sec. IV-C) including an online demonstration where the map is used for route following (Sec. IV-D). A demonstration video can be found at: <https://ori.ox.ac.uk/lidar-slam>.

A. Verification Performance

We focus on the alignability metric α of the alignment risk prediction since it is our primary contribution. We computed α between consecutive point clouds of our outdoor experiment, which we discuss later in Sec. IV-C. As seen in Fig. 6 (Left), the alignability filter based on a k-d tree is substantially faster than the original filter. As noted in Sec. III-D, our approach is less dependent on the point cloud size, due to only using the plane centroids. Whereas, the original alignability filter fully depends on all the points of the segments, resulting in higher computation time. Having tested our alignability filter for the datasets taken from Velodyne VLP-16, the average computation time is less than 0.5 seconds (almost 15x improvement) which is suitable for real-time operation.

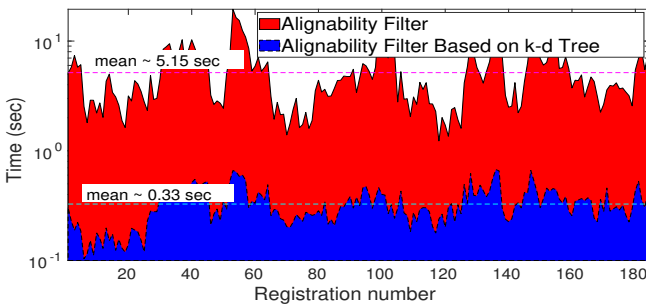


Fig. 5: Demonstration of the learning-based loop closure in the outdoor experiment.

Fig. 6 (Right) shows that our approach highly correlates with the result from the original approach. Finally, we refer the reader to the original work that provided a thorough comparison of alignment risk against Inverse Condition Number (ICN) [30] and Degeneracy parameter [31] in ill-conditioned scenarios. The former mathematically assessed the condition of the optimization, whereas the latter determines the degenerate dimension of the optimization.

B. Evaluation of Learned Loop Closure Detector

In the next experiment we explore the performance of the different loop closure methods using a dataset collected outdoors at the Fire Service College, Moreton-in-Marsh, UK.

1) *Robustness to viewpoint variation:* Fig. 5 shows a preview of our SLAM system with the learned loop closure method. We selected just two point clouds to create the map, with their positions indicated by the blue arrows in Fig. 5.

The robot executed two runs in the environment, which comprised 249 point clouds. We deliberately chose to traverse an offset path (red) the second time so as to determine how robust our algorithm is to translation and viewpoint variation. In total 50 loop closures were detected (green arrows) around the two map point clouds. Interestingly, the approach not only detected loop closures from both trajectories, with translational offsets up to 6.5 m, but also with orientation variation up to 180° - something not achievable by standard visual localization - a primary motivation for using LiDAR. Across the 50 loop closures an average of 5.24 segments were recognized per point cloud. The computed transformation had an Root Mean Square Error (RMSE) of 0.08 ± 0.02 m from the ground truth alignment. This was achieved in approximately 486 ms per query point cloud.

2) *Computation Time:* Fig. 8 shows a graph of the computation time. The computation time for the geometric loop closure method depends on the number of traversals

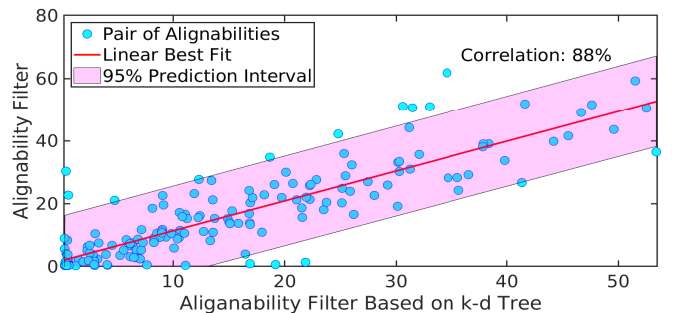


Fig. 6: Comparison of our proposed verification method with the original in terms of computation time (left) and performance (right).

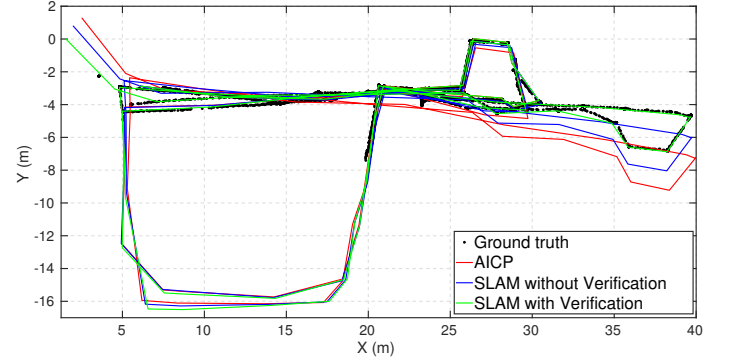
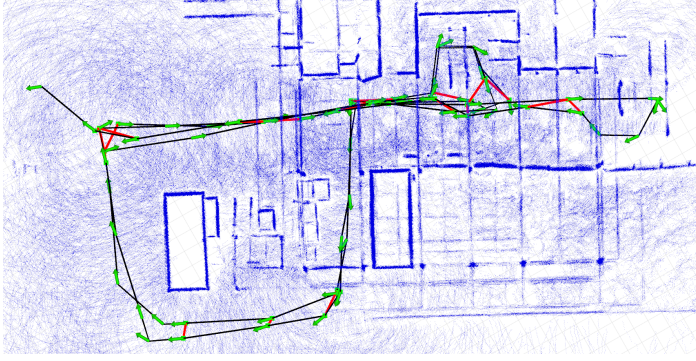


Fig. 7: Result of SLAM system with verification enabled (left). Estimated trajectories of algorithm variants versus ground truth (right).

around the same area. The geometric loop closures iterated over the nearest N clouds, based on a radius; the covariance and distance traveled caused it to slow down. Similarly, the verification method needs to iterate over large proportion of the point clouds in the same area, affecting the real time operation.

Instead, the learning loop closure proposal scales better with map size. It compares low dimensional feature descriptor vectors, which is much faster than the thousands of data points in Euclidean space.

C. Indoor and Outdoor Experiments

To evaluate the complete SLAM system, the robot walked indoor and outdoor along trajectories with the length of about 100 m and 250 m, respectively. Each experiment lasted about 45 minutes. Fig. 1 (Bottom) and Fig. 4 illustrate the test locations: industrial buildings. For quality evaluation, we compared the SLAM system, AICP, and the legged odometry (TSIF) using ground truth. As shown in Fig. 4, we used a Leica TS16 to automatically track a 360° prism rigidly mounted on top of the robot. This way, we managed to record the robot's position with millimeter accuracy at about 7 Hz (when in line of sight).

For evaluation metrics, we use Relative Pose Error (RPE) and Absolute Trajectory Error (ATE) [32]. RPE determines the regional accuracy of the trajectory over time. ATE is the RMSE of the Euclidean distance between the estimated trajectory and the ground truth.

As seen in Fig. 7 (Right), our SLAM system with verification is almost complete consistent with the ground truth. The verification algorithm approved 27 true-positive loop-closure factors (indicated in red in Fig. 7 (Left)) which were added to the factor graph. Without this verification 38 loop-closures were created, some in error, resulting in an inferior map.

Tab. II reports SLAM results with and without verification compared to the AICP LiDAR odometry and the TSIF legged state estimator. As the Leica TS16 does not provide rotational

Method	Translational Error (RMSE) (m)	Heading Error (RMSE) (deg)	Relative Pose Error (m)
SLAM with Verification	0.06	N/A	0.090
SLAM without Verification	0.23	1.6840	0.640
AICP (LiDAR odometry)	0.62	3.1950	1.310
TSIF (legged odometry)	5.40	36.799	13.64

TABLE II: Comparison of the localization accuracy for the different approaches.

estimates, we took the best performing method - SLAM with verification - and compared the rest of the trajectories to it with the ATE metric. Based on this experiment, the drift of SLAM with verification is less than 0.07%, satisfying many location-based tasks of the robot.

D. Experiments on the ANYmal

In a final experiment, we tested the SLAM system online on the ANYmal. After building a map with several loops (while teleoperated), we queried a path back to the operator station. Using the Dijkstra's algorithm [33], the shortest path was created using the factor graph. As each edge has previously been traverse, following the return trajectory returned the robot to the starting location. The supplementary video demonstrates the experiment.

V. CONCLUSION AND FUTURE WORK

This paper presented an accurate and robust LiDAR-SLAM system on a resource constrained legged robot using a factor graph-based optimization. We introduced an improved registration verification algorithm capable of running in real time. In addition, we leveraged a state-of-the-art learned loop closure detector which is sufficiently efficient to run online and had significant viewpoint robustness. We examined our system in indoor and outdoor industrial environments with a final demonstration showing online operation of the system on our robot. We will speed up our ICP registration to increase the update frequency in our SLAM system in future work and examine our system in more varied scenarios.

VI. ACKNOWLEDGMENT

We would like to thank our colleagues in ORI, especially Simona Nobili, for their help in this work. This research was supported by the Innovate UK-funded ORCA Robotics Hub (EP/R026173/1) and the EU H2020 Project MEMMO. M. Fallon is supported by a Royal Society University Research Fellowship.

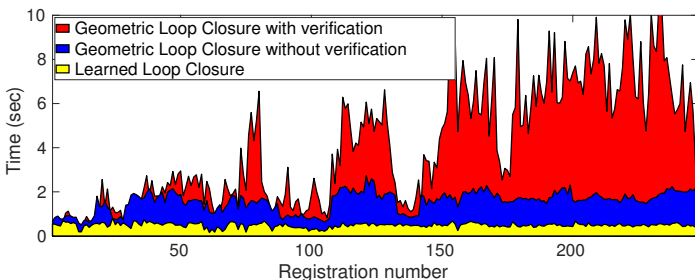


Fig. 8: Computation times of the considered loop closure methods.

REFERENCES

- [1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *TRO*, 2015.
- [2] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *ECCV*, 2014.
- [3] H. Porav, W. Maddern, and P. Newman, "Adversarial training for adverse conditions: Robust metric localisation using appearance transfer," in *ICRA*, 2018.
- [4] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–606.
- [5] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based ICP tuning for robust localization of a humanoid robot," in *ICRA*, 2017.
- [6] M. Bloesch, M. Hutter, M. A. Hoepffinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart, "State estimation for legged robots-consistent fusion of leg kinematics and IMU," in *RSS*, 2013.
- [7] O. Stasse, A. Davison, R. Sellaouti, and K. Yokoi, "Real-time 3D slam for humanoid robot considering pattern generator information," in *IROS*, 2006.
- [8] N. Kwak, O. Stasse, T. Foissotte, and K. Yokoi, "3D grid and particle based slam for a humanoid robot," in *Humanoids*, 2009.
- [9] G. Oriolo, A. Paolillo, L. Rosa, and M. Vendittelli, "Vision-based odometric localization for humanoids using a kinematic EKF," in *Humanoids*, 2012.
- [10] R. Wagner, U. Frese, and B. Bäuml, "Graph SLAM with signed distance function maps on a humanoid robot," in *IROS*, 2014.
- [11] R. Scona, S. Nobili, Y. R. Petillot, and M. Fallon, "Direct visual SLAM fusing proprioception for a humanoid robot," in *IROS*, 2017.
- [12] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *ISMAR*, 2011.
- [13] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, "ElasticFusion: Dense SLAM without a pose graph." *RSS*, 2015.
- [14] M. Milford and G. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *ICRA*, 2012.
- [15] H.-W. Park, P. Wensing, and S. Kim, "Online planning for autonomous running jumps over obstacles in high-speed quadrupeds," in *RSS*, 2015.
- [16] S. Nobili, M. Camurri, V. Barasuol, M. Focchi, D. Caldwell, C. Semini, and M. Fallon, "Heterogeneous sensor fusion for accurate state estimation of dynamic legged robots," in *RSS*, 2017.
- [17] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in real-time," in *RSS*, 2014.
- [18] J.-E. Deschaud, "Imls-slam: scan-to-model matching based on 3d data," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2480–2485.
- [19] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *IROS*, 2018.
- [20] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the bayes tree," *IJRR*, 2012.
- [21] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments," in *Robotics: Science and Systems*, 2018.
- [22] R. Dubé, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "An online multi-robot SLAM system for 3D LiDARS," in *IROS*, 2017.
- [23] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "Segmatch: Segment based place recognition in 3D point clouds," in *ICRA*, 2017.
- [24] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The two-state implicit filter recursive estimation for mobile robots," *RAL*, 2017.
- [25] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing icp variants on real-world data sets," *Autonomous Robots*, 2013.
- [26] G. Tinchev, A. Penate-Sanchez, and M. Fallon, "Learning to see the wood for the trees: Deep laser localization in urban and natural environments on a CPU," *RAL*, 2019.
- [27] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *NIPS*, 2018.
- [28] S. Nobili, G. Tinchev, and M. Fallon, "Predicting alignment risk to prevent localization failure," in *ICRA*, 2018.
- [29] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, and M. Bloesch, "Anymal - a highly mobile and dynamic quadrupedal robot," in *IROS*, 2016.
- [30] E. Cheney and D. Kincaid, *Numerical mathematics and computing*. Cengage Learning, 2012.
- [31] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *ICRA*, 2016.
- [32] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *IROS*, 2012.
- [33] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, 1959.

Summary

This chapter incorporated the proposed global localization system from chapter 4 as a loop closure detector into a SLAM system that run on the ANYmal quadruped robot in an industrial setting. It also presented an improvement to the alignment risk prediction framework from chapter 7 allowing it to run in real time.

A key observation of this chapter confirmed that the proposed system from chapter 4 is desirable for detecting geometric loop closures, due to its scalability to large maps. This is crucial for large-scale experiments, where the prior maps were comprised of hundreds of kilometers and multiple loops around the same area, such as the case of the Oxford RobotCar dataset [Maddern et al. 2017]. Another interesting finding was the ability of the network to generalize to completely unseen industrial data. This suggested that the proposed system could be deployed for exploration tasks.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

Title of Paper	Online 3D LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Ramezani, M., Tinchev, G., Iuganov E., & Fallon, M. (2020, May). Online LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure. In <i>2020 IEEE International Conference on Robotics and Automation (ICRA)</i> (pp. 4158-4164). IEEE.

Student Confirmation

Student Name:	Georgi Tinchev		
Contribution to the Paper	- Collaborated with the main author to improve the verification method's runtime - Analysed computational efficiency of the verification method - Designed the experiments involving the deep learned loop closure - Compared the proposed loop closure to a geometric loop closure baseline - Wrote and corrected parts of the manuscript		
		Date	30/12/2020

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Dr. Maurice Fallon		
Supervisor comments: The contributions listed above are accurate		
	Date	30/12/2020

This completed form should be included in the thesis, at the end of the relevant chapter.

9

Future work & Conclusions

In the thesis we have addressed different topics ranging from loop closures in robotics to keypoint detection and feature learning in computer vision. In the following chapter we outline future research directions.

Multi season localization

A possible extension of this work would also consider evaluating this system during multiple seasons. While camera relocalization is more explored [Sattler et al. 2018], LIDAR place recognition during various seasons is less researched. Specifically for natural environments, different seasons can alter the appearance of trees, bushes, and canopy drastically. For example, while cameras might struggle recognizing scenes with snow, LiDAR has the potential to overcome these difficulties implicitly.

Segmentation Improvements

A drawback of the system in chapters 3 to 5 is the used predefined segmentation for both natural and urban scenarios. In our observations euclidean segmentation works well for both, however, in some scenarios, where there is no separability between individual objects it could cause problems. Such an example is when traversing indoor environments or when traversing a closed space between two buildings. Therefore, further research could potentially look

into (non) semantic segmentation of objects in different environments. Another potential improvement to the system would be to merge extracted segments over time and considering multiple segments over a certain time period for matching at a specific location. This would be useful when very long objects are observed, for example buses or long continuous hedges.

Multi-robot localization

An interesting extension of the presented work would be to develop a collaborative multi-robot localization system with a common database for storing and matching segments. One of the potential advantages of the proposed system is the small 16 dimensional descriptor vector per segment. This would result in at least 16 times compression of data that is to be transferred over a network, which would require minimal amount of bandwidth. In a multi-robot collaborative exploration tasks it is important to minimize the number of messages sent between robots and their size. This has a direct application to real-world tasks for autonomous driving where vehicles collaborate to explore a previously unknown environment.

3D reconstructions of segments

The proposed approach also opens the doors to more research in 3D reconstruction. Specifically, a potential application is creating a reconstruction network that is able to recreate the segments and display the results to a separate visualizing computer, while a robot is operating in real time. This would efficiently create a visualization for an expert in the field to monitor the progress of the system. While similar works have been done before by Dubé et al. [2019], they assume semantic knowledge of the segments. Lifting this limitation is still a major challenge in the field.

Learning coarse geometric alignment

Another area for improvement would be the registration verification module. Chapter 4 only depended on the probability extracted from the description

network and the geometry of objects in the scene to generate a 6 DoF transformation. A more complex strategy would be to consider continuously growing cliques of objects in the scene, as well as refinement of the original keypoint per segment. A recent developments in deep learning methods such as the works of Lu et al. [2019] and Wang and Solomon [2019] have identified differentiable SVD decomposition methods that can generate a transformation reliably in just a single forward pass, without the need of iteratively solve the model fitting problem. A recent alternative to these methods are the ones presented by Zhou et al. [2016] and Gojcic et al. [2020] where the global and local registration problems are tackled together in a single optimization.

Multi-experience LIDAR segment SLAM

A plausible extension of this system could consider multiple experiences when navigating. For example, a construction site would change quite often as more of the core structure is completed. This would inevitably affect the underneath SLAM system if one did not consider the evolution of the surrounding objects. A possible solution to this would be to inspect the environment multiple times and keep a record for each extracted segment. Further research could explore how these multiple objects can be merged or used for long-term localization.

Improved learning for LIDAR localization

A growth area for further research would be to implement rotation invariance within the network architecture and discarding the currently needed pre-processing segment alignment step. A suitable extension would be to consider the Rigorous-Rotation Invariant features that can be implemented instead of the relative distances between points in PointCNN [Chen et al. 2019]. Another interesting area would be to explore different objective functions, like the quadruplet loss [Angelina Uy and Hee Lee 2018], or the n-tuple loss [Deng et al. 2018] to see if the feature space could be further improved. Furthermore, it would be interesting to see if our strategy to improve performance of PointCNN is comparable to the optimized model of PointVoxel [Liu et al. 2019] and even

leverage the proposed sub-sampling method for their model in hope for further computation time reduction.

Real-time Keypoint Extraction

A drawback of the system proposed in chapter 6 is that the extracted saliency information requires the gradients of a specific layer with respect to the input. This was a costly operation since running backpropagation during inference is very inefficient, even for a few layers of the architecture. An interesting forward direction would be to have a second, smaller network to predict the gradient response of the given descriptor model with a certain input. This could be interpreted as a parallel to a teacher-student architecture used in generative modeling [Oord et al. 2018]. This would allow the system to run in real time without the need to compute gradients of the original descriptor network. As an application result, this would alleviate the need for environment-specific segmentation methods and allow a unified framework to work in both scenarios. It would then be interesting to compare the performance of segment-based methods against the proposed keypoint extraction and segment-generation strategy. A further research direction could explore the underlying meaning of saliency and why the gradient of the learned manifold is meaningful without explicit supervision.

End-to-end segment-based learning

Another interesting research direction would be to combine the individual differentiable modules into a single end-to-end learning approach. This has been done in the work of Yew and Lee [2018], however, the training procedure is complicated with the need of pre-training the descriptor. The end-to-end training could include the keypoint extraction stage, keypoint association, description, as well as a learned registration strategy to produce a pose in the prior map. This could significantly reduce the computation time, since a single forward pass of the end-to-end learned network is required, bringing down the total necessary time for processing a single point cloud.

9.1 Conclusion

This thesis presents state-of-the-art localization system capable of operating in both urban and natural environments. We chose a segment-based approach to localization where we split the task in four modules - segmentation, description, matching, and geometric registration. Initially, we have created a hand-crafted feature descriptor that is suitable for both urban and natural scenes. We noticed that while this closes the performance gap between the two environments, there is still room for improvement. We have then turned to deep learning to learn a feature space representation and created an approach capable of running inference on a CPU. We trained this network using a combination of triplet and pairwise losses that learned to cluster matching from non-matching segments in the data manifold. We discovered that we can further push the limits of what the proposed network is capable of in terms of computational complexity by sub-sampling in the inner layers of the architecture. We then improved the segmentation module of the pipeline by proposing to extract keypoints from an already trained differentiable descriptor. Finally, we looked at two applications of this thesis - one where we can prevent localization failures from happening by using the geometry of the scene, and an online SLAM method that uses the proposed method as a loop closure proposal while operating online on the ANYmal quadrupedal robot. We hope that this thesis will prompt broader research that looks into using segment-based localization for long-term autonomy in challenging terrains.

References

- Agarwal, S., Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski [2011]. "Building Rome in a day". In: *Communications of the ACM* 54.10, pp. 105–112.
- Aldera, R., D. De Martini, M. Gadd, and P. Newman [May 2019]. "Fast radar motion estimation with a learnt focus of attention using weak supervision". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 1190–1196.
- Angelina Uy, M. and G. Hee Lee [2018]. "PointNetVLAD: deep point cloud based retrieval for large-scale place recognition". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4470–4479.
- Arandjelovic, R., P. Gronat, A. Torii, T. Pajdla, and J. Sivic [2016]. "NetVLAD: CNN architecture for weakly supervised place recognition". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 5297–5307.
- Arun, K. S., T. S. Huang, and S. D. Blostein [1987]. "Least-squares fitting of two 3-D point sets". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5, pp. 698–700.
- Azimi, S., B. Lall, and T. K. Gandhi [2019]. "Performance evaluation of 3D keypoint detectors and descriptors for plants health classification". In: *International Conference on Machine Vision Applications*. IEEE, pp. 1–6.
- Baldwin, I. and P. Newman [2012]. "Laser-only road-vehicle localization with dual 2D push-broom LIDARs and 3D priors". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 2490–2497.
- Bay, H., T. Tuytelaars, and L. Van Gool [2006]. "SURF: speeded up robust features". In: *Proc. ECCV*. Springer, pp. 404–417.
- Benbihi, A., M. Geist, and C. Pradalier [2019]. "ELF: embedded localisation of features in pre-trained CNN". In: *Proc. ICCV*, pp. 7940–7949.
- Besl, P. J. and N. D. McKay [1992]. "Method for registration of 3-D shapes". In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. International Society for Optics and Photonics, pp. 586–606.
- Biber, P. and W. Straßer [2003]. "The normal distributions transform: a new approach to laser scan matching". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. Vol. 3. IEEE, pp. 2743–2748.
- Bosse, M. and R. Zlot [2013]. "Place recognition using keypoint voting in large 3D LIDAR datasets". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 2677–2684.
- Cadena, C., L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard [2016]. "Past, present, and future of simultaneous localization and mapping: toward the robust-perception age". In: *IEEE Transactions on Robotics* 32.6, pp. 1309–1332.
- Cadena, C., A. Dick, and I. D. Reid [2015]. "A fast, modular scene understanding system using context-aware object detection". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 4859–4866.

- Calonder, M., V. Lepetit, C. Strecha, and P. Fua [2010]. "BRIEF: binary robust independent elementary features". In: *Proc. ECCV*. Springer, pp. 778–792.
- Campos, C., R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós [2020]. "ORB-SLAM3: an accurate open-source library for visual, visual-inertial and multi-map SLAM". In: *arXiv preprint arXiv:2007.11898*.
- Castle, R. O., D. J. Gawley, G. Klein, and D. W. Murray [2007]. "Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 4102–4107.
- Cen, S. and P. Newman [May 2019]. "Radar-only ego-motion estimation in difficult settings via graph matching". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 298–304.
- Chen, C., G. Li, R. Xu, T. Chen, M. Wang, and L. Lin [2019]. "ClusterNet: deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4994–5002.
- Chen, Y. and G. Medioni [1992]. "Object modelling by registration of multiple range images". In: *Image and vision computing* 10.3, pp. 145–155.
- Cheney, E. W. and D. R. Kincaid [2012]. *Numerical mathematics and computing*. Cengage Learning.
- Chollet, F. [2017]. "Xception: deep learning with depthwise separable convolutions". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1251–1258.
- Choy, C. B., D. Xu, J. Gwak, K. Chen, and S. Savarese [2016]. "3D-R2N2: a unified approach for single and multi-view 3D object reconstruction". In: *Proc. ECCV*, pp. 628–644.
- Churchill, W. and P. Newman [2013]. "Experience-based navigation for long-term localisation". In: *Intl. J. of Robotics Research*.
- Civera, J., D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. M. M. Montiel [2011]. "Towards semantic SLAM using a monocular camera". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 1277–1284.
- Cramariuc, A., R. Dubé, H. Sommer, R. Siegwart, and I. Gilitschenski [2018]. "Learning 3D segment descriptors for place recognition". In: *arXiv preprint arXiv:1804.09270*.
- Cummins, M. and P. Newman [2008]. "FAB-MAP: probabilistic localization and mapping in the space of appearance". In: *Intl. J. of Robotics Research* 27.6, pp. 647–665.
- Davison, A. J., I. D. Reid, N. D. Molton, and O. Stasse [2007]. "MonoSLAM: real-time single camera SLAM". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.6, pp. 1052–1067.
- Deng, H., T. Birdal, and S. Ilic [2018]. "PPFNet: global context aware local features for robust 3D point matching". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 195–205.
- Dissanayake, M. W. M. G., P. Newman, S. Clark, H. F. Durrant-whyte, and M. Csorba [2001]. "A solution to the simultaneous localization and map building (SLAM) problem". In: *IEEE Transactions on Robotics* 17, pp. 229–241.
- Dosovitskiy, A., P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox [Dec. 2015]. "FlowNet: learning optical flow with convolutional networks". In: *Proc. ICCV*.

- Douillard, B., A. Quadros, P. Morton, J. P. Underwood, M. D. Deuge, S. Hugosson, M. Hallström, and T. Bailey [May 2012]. "Scan segments matching for pairwise 3D alignment". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 3033–3040.
- Dubé, R., A. Cramariuc, D. Dugas, J. Nieto, R. Siegwart, and C. Cadena [June 2018]. "SegMap: 3D segment mapping using data-driven descriptors". In: *Robotics: Science and Systems (RSS)*. Pittsburgh, Pennsylvania, pp. 20–30.
- Dubé, R., A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena [2019]. "SegMap: segment-based mapping and localization using data-driven descriptors". In: *Intl. J. of Robotics Research* 0.0.
- Dubé, R., D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena [2017]. "SegMatch: segment based place recognition in 3D point clouds". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 5266–5272.
- Eigen, D., C. Puhersch, and R. Fergus [2014]. "Depth map prediction from a single image using a multi-scale deep network". In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2366–2374.
- Elbaz, G., T. Avraham, and A. Fischer [2017]. "3D point cloud registration for localization using a deep neural network auto-encoder". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4631–4640.
- Engel, J., T. Schöps, and D. Cremers [2014]. "LSD-SLAM: large-scale direct monocular SLAM". In: *Proc. ECCV*. Springer, pp. 834–849.
- Engelcke, M., D. Rao, D. Z. Wang, C. H. Tong, and I. Posner [2017]. "Vote3Deep: fast object detection in 3D point clouds using efficient convolutional neural networks". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 1355–1361.
- Fallon, M. F., J. Folkesson, H. McClelland, and J. J. Leonard [July 2013]. "Relocating underwater features autonomously using sonar-based SLAM". In: *Journal of Oceanic Engineering* 38 [3], pp. 500–513.
- Fischler, M. A. and R. C. Bolles [1981]. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6, pp. 381–395.
- Flitton, G. T., T. P. Breckon, and N. M. Bouallagu [2010]. "Object recognition using 3D SIFT in complex CT volumes." In: *Proc. BMVC*. Vol. 1, pp. 1–12.
- Furgale, P. T. and T. D. Barfoot [2010]. "Visual teach and repeat for long-range rover autonomy". In: *Journal of Field Robotics* 27.5, pp. 534–560.
- Gawel, A., T. Cieslewski, R. Dubé, M. Bosse, R. Siegwart, and J. Nieto [2016]. "Structure-based vision-laser matching". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 182–188.
- Gawel, A., R. Dubé, H. Surmann, J. Nieto, R. Siegwart, and C. Cadena [2017]. "3D registration of aerial and ground robots for disaster response: an evaluation of features, descriptors, and transformation estimation". In: *IEEE International Symposium on Safety, Security and Rescue Robotics*. IEEE, pp. 27–34.
- Geiger, A., P. Lenz, C. Stiller, and R. Urtasun [2013]. "Vision meets robotics: the KITTI dataset". In: *Intl. J. of Robotics Research* 32.11, pp. 1231–1237.
- Girshick, R., J. Donahue, T. Darrell, and J. Malik [2014]. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 580–587.
- Gojcic, Z., C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal [2020]. "Learning multiview 3D point cloud registration". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1759–1769.

- Granström, K., T. B. Schön, J. I. Nieto, and F. T. Ramos [2011]. "Learning to close loops from range data". In: *Intl. J. of Robotics Research* 30.14, pp. 1728–1754.
- Haoqiang, F., H. Su, and L. J. Guibas [2017]. "A point set generation network for 3D object reconstruction from a single image". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2463–2471. URL: <https://doi.org/10.1109/CVPR.2017.264>.
- Harris, C. G. and M. Stephens [1988]. "A combined corner and edge detector". In: *Alvey vision conference*. Vol. 15. 50. Citeseer, pp. 10–5244.
- Hartley, R. and A. Zisserman [2003]. *Multiple view geometry in computer vision*. Cambridge university press.
- He, K., X. Zhang, S. Ren, and J. Sun [2016]. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 770–778.
- Hess, W., D. Kohler, H. Rapp, and D. Andor [May 2016]. "Real-time loop closure in 2D LIDAR SLAM". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 1271–1278.
- Holz, D., A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke [2015]. "Registration with the point cloud library: a modular framework for aligning in 3-D". In: *IEEE Robotics & Automation Magazine* 22.4, pp. 110–124.
- Hover, F., R. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. Leonard [Oct. 2012]. "Advanced perception, navigation and planning for autonomous in-water ship hull inspection". In: *Intl. J. of Robotics Research* 31.12, pp. 1445–1464.
- Hu, G., S. Huang, L. Zhao, A. Alempijevic, and G. Dissanayake [2012]. "A robust RGB-D SLAM algorithm". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 1714–1719.
- Jiang, M., Y. Wu, T. Zhao, Z. Zhao, and C. Lu [2018]. "PointSIFT: a SIFT-like network module for 3D point cloud semantic segmentation". In: *arXiv preprint arXiv:1807.00652*.
- Krizhevsky, A., I. Sutskever, and G. E. Hinton [2012]. "ImageNet classification with deep convolutional neural networks". In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1097–1105.
- Leutenegger, S., M. Chli, and R. Y. Siegwart [2011]. "BRISK: binary robust invariant scalable keypoints". In: *Proc. ICCV*. IEEE, pp. 2548–2555.
- Levinson, J., M. Montemerlo, and S. Thrun [2007]. "Map-based precision vehicle localization in urban environments". In: *Robotics: Science and Systems (RSS)*, pp. 110–118.
- Li, B. [2017]. "3D fully convolutional network for vehicle detection in point cloud". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 1513–1518.
- Li, G., M. Muller, A. Thabet, and B. Ghanem [2019]. "DeepGCNs: can GCNs go as deep as CNNs?" In: *Proc. ICCV*, pp. 9267–9276.
- Li, J. and G. H. Lee [2019]. "USIP: unsupervised stable interest point detection from 3D point clouds". In: *Proc. ICCV*, pp. 361–370.
- Li, Y., R. Bu, M. Sun, W. Wu, X. Di, and B. Chen [2018]. "PointCNN: convolution on x-transformed points". In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 820–830.
- Liu, H., F. Sun, B. Fang, and X. Zhang [Jan. 2017]. "Robotic room-level localization using multiple sets of sonar measurements". In: *IEEE Transactions on Instrumentation and Measurement* 66.1, pp. 2–13.

- Liu, Z., H. Tang, Y. Lin, and S. Han [2019]. "Point-Voxel CNN for efficient 3D deep learning". In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 965–975.
- Lowe, D. G. [2004]. "Distinctive image features from scale-invariant keypoints". In: *International Journal of Computer Vision* 60.2, pp. 91–110.
- Lu, W., G. Wan, Y. Zhou, X. Fu, P. Yuan, and S. Song [2019]. "DeepVCP: an end-to-end deep neural network for point cloud registration". In: *Proc. ICCV*, pp. 12–21.
- Maddern, W., G. Pascoe, C. Linegar, and P. Newman [2017]. "1 year, 1000 km: the Oxford robotcar dataset". In: *Intl. J. of Robotics Research* 36.1, pp. 3–15.
- Magnusson, M., H. Andreasson, A. Nüchter, and A. J. Lilienthal [2009]. "Automatic appearance-based loop detection from three-dimensional laser data using the normal distributions transform". In: *Journal of Field Robotics* 26.11-12, pp. 892–914.
- Magnusson, M., A. Lilienthal, and T. Duckett [2007]. "Scan registration for autonomous mining vehicles using 3D-NDT". In: *Journal of Field Robotics* 24.10, pp. 803–827.
- McManus, C., B. Upcroft, and P. Newman [July 2014]. "Scene signatures: localised and point-less features for localisation". In: *Robotics: Science and Systems (RSS)*. Berkeley, CA, USA, pp. 177–185.
- Milford, M. J. and G. F. Wyeth [2012]. "SeqSLAM: visual route-based navigation for sunny summer days and stormy winter nights". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 1643–1649.
- Mur-Artal, R., J. M. M. Montiel, and J. D. Tardos [2015]. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE Transactions on Robotics* 31.5, pp. 1147–1163.
- Newcombe, R. A., S. J. Lovegrove, and A. J. Davison [2011]. "DTAM: dense tracking and mapping in real-time". In: *Proc. ICCV*. IEEE, pp. 2320–2327.
- Ni, D., Y. Qu, X. Yang, Y. P. Chui, T.-T. Wong, S. S. Ho, and P. A. Heng [2008]. "Volumetric ultrasound panorama based on 3D SIFT". In: *Proc. MICCAI*. Springer, pp. 52–60.
- Nieto, J., T. Bailey, and E. Nebot [2006]. "Scan-SLAM: combining EKF-SLAM and scan correlation". In: *Field and Service Robotics*. Springer, pp. 167–178.
- Nobili, S., R. Scona, M. Caravagna, and M. Fallon [2017]. "Overlap-based ICP tuning for robust localization of a humanoid robot". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 4721–4728.
- Nobili, S., G. Tinchev, and M. Fallon [May 2018]. "Predicting alignment risk to prevent localization failure". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 1003–1010.
- Nocedal, J. and S. Wright [2006]. *Numerical optimization*. Springer Science & Business Media.
- Nüchter, A., K. Lingemann, J. Hertzberg, and H. Surmann [2007]. "6D SLAM—3D mapping outdoor environments". In: *Journal of Field Robotics* 24.8-9, pp. 699–722.
- Oord, A., Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. Driessche, E. Lockhart, L. Cobo, F. Stimberg, et al. [2018]. "Parallel WaveNet: fast high-fidelity speech synthesis". In: *Proc. ICML*. PMLR, pp. 3918–3926.
- Oquab, M., L. Bottou, I. Laptev, and J. Sivic [2014]. "Learning and transferring mid-level image representations using convolutional neural networks". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1717–1724.
- Park, Y. S., J. Jeong, Y. Shin, and A. Kim [May 2019a]. "Radar dataset for robust localization and mapping in urban environment". In: *Montreal*, pp. 66–70.

- Park, Y. S., J. Kim, and A. Kim [Nov. 2019b]. "Radar localization and mapping for indoor disaster environments via multi-modal registration to prior LiDAR map". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. Accepted To appear. Macau, pp. 1–9.
- Pizzoli, M., C. Forster, and D. Scaramuzza [2014]. "REMODE: probabilistic, monocular dense reconstruction in real time". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 2609–2616.
- Pomerleau, F., F. Colas, R. Siegwart, and S. Magnenat [2013]. "Comparing ICP variants on real-world data sets". In: *Autonomous Robots* 34.3, pp. 133–148.
- Porav, H., W. Maddern, and P. Newman [May 2018]. "Adversarial training for adverse conditions: robust metric localisation using appearance transfer". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 1011–1018.
- Qi, C. R., W. Liu, C. Wu, H. Su, and L. J. Guibas [2018]. "Frustum PointNets for 3D object detection from RGB-D data". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 918–927.
- Qi, C. R., H. Su, K. Mo, and L. J. Guibas [2017a]. "PointNet: deep learning on point sets for 3D classification and segmentation". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 652–660.
- Qi, C. R., L. Yi, H. Su, and L. J. Guibas [2017b]. "PointNet++: deep hierarchical feature learning on point sets in a metric space". In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5099–5108.
- Ramezani, M., G. Tinchev, E. Iuganov, and M. Fallon [June 2020]. "Online LiDAR-SLAM for legged robots with robust registration and deep-learned loop closure". In: *Proc. Intl. Conf. on Robotics and Automation*.
- Ren, S., K. He, R. Girshick, and J. Sun [2015]. "Faster R-CNN: towards real-time object detection with region proposal networks". In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 91–99.
- Rister, B., M. A. Horowitz, and D. L. Rubin [2017]. "Volumetric image registration from invariant keypoints". In: *IEEE Transactions on Image Processing* 26.10, pp. 4900–4910.
- Röhling, T., J. Mack, and D. Schulz [2015]. "A fast histogram-based similarity measure for detecting loop closures in 3-D LIDAR data". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 736–741.
- Rosten, E. and T. Drummond [2006]. "Machine learning for high-speed corner detection". In: *Proc. ECCV*. Springer, pp. 430–443.
- Rublee, E., V. Rabaud, K. Konolige, and G. Bradski [2011]. "ORB: an efficient alternative to SIFT or SURF". In: *Proc. ICCV*. IEEE, pp. 2564–2571.
- Rusu, R. B., N. Blodow, and M. Beetz [2009]. "Fast point feature histograms (FPFH) for 3D registration". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 3212–3217.
- Rusu, R. B. and S. Cousins [2011]. "3D is here: point cloud library (PCL)". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 1–4.
- Sattler, T., W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, et al. [2018]. "Benchmarking 6DoF outdoor visual localization in changing conditions". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 8601–8610.
- Scovanner, P., S. Ali, and M. Shah [2007]. "A 3-dimensional SIFT descriptor and its application to action recognition". In: *Proceedings of the 15th ACM international conference on Multimedia*, pp. 357–360.

- Segal, A., D. Haehnel, and S. Thrun [2009]. "Generalized-ICP". In: *Robotics: Science and Systems (RSS)*. Vol. 2. 4. Seattle, WA, p. 435.
- Serafin, J. and G. Grisetti [2015]. "NICP: dense normal based point cloud registration". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 742–749.
- Shan, T. and B. Englot [2018]. "LeGO-LOAM: lightweight and ground-optimized LIDAR odometry and mapping on variable terrain". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 4758–4765.
- Shan, T., B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus [2020]. "LIO-SAM: tightly-coupled LIDAR inertial odometry via smoothing and mapping". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*.
- Sharif Razavian, A., H. Azizpour, J. Sullivan, and S. Carlsson [2014]. "CNN features off-the-shelf: an astounding baseline for recognition". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition Workshops*, pp. 806–813.
- Simo-Serra, E., E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer [Dec. 2015]. "Discriminative learning of deep convolutional feature point descriptors". In: *Proc. ICCV*, pp. 118–126.
- Simonyan, K. and A. Zisserman [2014]. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556*.
- Steder, B., G. Grisetti, and W. Burgard [2010a]. "Robust place recognition for 3D range data based on point features". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 1400–1405.
- Steder, B., M. Ruhnke, S. Grzonka, and W. Burgard [2011]. "Place recognition in 3D scans using a combination of bag of words and point feature based relative pose estimation". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE, pp. 1249–1255.
- Steder, B., R. B. Rusu, K. Konolige, and W. Burgard [2010b]. "NARF: 3D range image features for object recognition". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems Workshops*. Vol. 44.
- Thrun, S., D. Fox, W. Burgard, and F. Dellaert [1999]. "Monte carlo localization for mobile robots". In: *Proc. Intl. Conf. on Robotics and Automation*, pp. 1322–1328.
- Tinchev, G., S. Nobili, and M. Fallon [2018]. "Seeing the wood for the trees: reliable localization in urban and natural environments". In: *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*, pp. 8239–8246.
- Tinchev, G., A. Penate-Sanchez, and M. Fallon [2019]. "Learning to see the wood for the trees: deep laser localization in urban and natural environments on a CPU". In: *IEEE Robotics and Automation Letters* 4.2, pp. 1327–1334.
- Trefethen, L. N. and D. Bau III [1997]. *Numerical linear algebra*. Vol. 50. Siam.
- Uijlings, J. R., K. E. Van De Sande, T. Gevers, and A. W. Smeulders [2013]. "Selective search for object recognition". In: *International Journal of Computer Vision* 104.2, pp. 154–171.
- Ulrich, I. and I. Nourbakhsh [2000]. "Appearance-based place recognition for topological localization". In: *Proc. Intl. Conf. on Robotics and Automation*. Vol. 2. IEEE, pp. 1023–1029.
- Verdie, Y., K. Yi, P. Fua, and V. Lepetit [2015]. "TILDE: A temporally invariant learned detector". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 5279–5288.

- Wang, J., Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu [2014]. "Learning fine-grained image similarity with deep ranking". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1386–1393.
- Wang, Y. and J. M. Solomon [2019]. "Deep closest point: learning representations for point cloud registration". In: *Proc. ICCV*, pp. 3523–3532.
- Wang, Y., Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon [2019]. "Dynamic graph CNN for learning on point clouds". In: *ACM Transactions On Graphics* 38.5, pp. 1–12.
- Wolcott, R. W. and R. M. Eustice [2017]. "Robust LIDAR localization using multiresolution Gaussian mixture maps for autonomous driving". In: *Intl. J. of Robotics Research* 36, pp. 292–319.
- Ye, H., Y. Chen, and M. Liu [2019]. "Tightly coupled 3D LIDAR inertial odometry and mapping". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 3144–3150.
- Yew, Z. J. and G. H. Lee [2018]. "3DFeat-Net: weakly supervised local 3D features for point cloud registration". In: *Proc. ECCV*. Springer, pp. 630–646.
- Yi, K. M., E. Trulls, V. Lepetit, and P. Fua [2016]. "LIFT: learned invariant feature transform". In: *Proc. ECCV*. Springer, pp. 467–483.
- Zach, C., A. Penate-Sanchez, and M.-T. Pham [2015]. "A dynamic programming approach for fast and robust object pose recognition from range images". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 196–203.
- Žbontar, J. and Y. LeCun [2016]. "Stereo matching by training a convolutional neural network to compare image patches". In: *The Journal of Machine Learning Research* 17.1, pp. 2287–2318.
- Zhang, J., M. Kaess, and S. Singh [2016]. "On degeneracy of optimization-based state estimation problems". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 809–816.
- Zhang, J. and S. Singh [2014]. "LOAM: LIDAR odometry and mapping in real-time". In: *Robotics: Science and Systems (RSS)*, pp. 49–57.
- [2015]. "Visual-LIDAR odometry and mapping: low-drift, robust, and fast". In: *Proc. Intl. Conf. on Robotics and Automation*. IEEE, pp. 2174–2181.
- Zhong, Y. [2009]. "Intrinsic shape signatures: a shape descriptor for 3D object recognition". In: *Proc. ICCV Workshops*. IEEE, pp. 689–696.
- Zhou, Q.-Y., J. Park, and V. Koltun [2016]. "Fast global registration". In: *Proc. ECCV*. Springer, pp. 766–782.
- Zhou, Y. and O. Tuzel [2018]. "VoxelNet: end-to-end learning for point cloud based 3D object detection". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4490–4499.
- Zhuang, Y., N. Jiang, H. Hu, and F. Yan [2012]. "3-D-laser-based scene measurement and place recognition for mobile robots in dynamic indoor environments". In: *IEEE Transactions on Instrumentation and Measurement* 62.2, pp. 438–450.